INTRO TO CYBER SECURITY - BASICS

Easy Cyber Sec - The Best Cybersecurity Training in your language

www.easycybersec.com

Phone: +91 - 9597723169
WhatsApp: +91 - 9791956182

www.easycybersec.com

Phone: +91 – 9597723169
WhatsApp: +91 – 9791956182

This book aims to introduce you to the basic concepts of ethical hacking & cybersecurity. As the title says, this is just an informational book, for more detailed contents and learning join us at **Easy Cyber Sec.**

Contents

Introduction

In 2004 we had e-commerce via websites; in 2019 we have smart phone apps, the Internet of Things, as well as an expanded use of e-commerce websites. Internet traffic is far more than just humorous YouTube videos or Facebook updates about our vacations. Now it is the heart and soul of commerce, both domestic and international. Internet communication even plays a central role in military operations and diplomatic relations. In addition to smart phones, we now have smart watches and even vehicles that have Wi-Fi hotspots and smart technology. Our lives are inextricably intertwined with the online world. We file our taxes online, shop for a home online, book our next vacation online, and even look for a date online. Because so much of our business is transacted online, a great deal of personal information is stored in computers. Medical records, tax records, school records, and more are all stored in computer databases.

This leads to some very important questions:

1. How is information safeguarded?
2. What are the vulnerabilities to these systems?
3. What steps are taken to ensure that these systems and data are safe?
4. Who can access my information?

Unfortunately, not only has technology and Internet access expanded, but so have the dangers. How serious is the problem? According to a 2014 article in *SC Magazine*,1 "Cyber-crime and economic espionage cost the global economy more than $445 billion annually, which a report from the Center for Strategic and International Studies, says puts cyber-crime on par with the economic impact of global drug trafficking." Another study looked at specific companies and the cost of cybercrime in 2013. That study reported, "We found that the average annualized cost of cyber-crime for 60 organizations in our study is $11.6 million per year, with a range of $1.3 million to $58 million. In 2012, the average annualized cost was $8.9 million. This represents an increase in cost of 26 percent or $2.6 million from the results of our cyber cost study published last year."

The situation is not improving, either. According to a Pricewaterhouse Coopers study, in 2015 38% more security incidents were detected than in 2014. The same study showed a 56% increase in theft of intellectual property. In spite of daily horror stories, however, many people (including some law enforcement professionals and trained computer professionals) lack an adequate understanding about the reality of these threats. Clearly the media will focus attention on the most dramatic computer security breaches, not necessarily giving an accurate picture of the most plausible threat scenarios. It is not uncommon to encounter the occasional system administrator whose knowledge of computer security is inadequate.

How Seriously Should You Take Threats to Network Security?

The first step in understanding computer and network security is to formulate a realistic assessment of the threats to those systems. You will need a clear picture of the dangers in order to adequately prepare a defense. There seem to be two extreme attitudes regarding computer security. The first group assumes there is no real threat. Subscribers to this belief feel that there is little real danger to computer systems and that much of the negative news is simply unwarranted panic. They often believe taking only minimal security precautions should ensure the safety of their systems. The prevailing sentiment is, if our organization has not been attacked so far, we must be secure. If decision makers subscribe to this point of view, they tend to push a reactive approach to security. They will wait to address security. The truly talented hacker is no more common than the truly talented concert pianist. Consider how many people take piano lessons at some point in their lives. Now consider how many of those ever truly become virtuosos. The same is true of computer hackers. Keep in mind that even those who do possess the requisite skills need to be motivated to expend the time and effort to compromise your system.

A better way to assess the threat level to your system is to weigh the attractiveness of your system to potential intruders against the security measures in place. Keep in mind, too, that the greatest external threat to any system is not hackers, but malware and denial of service (DoS) attacks. Malware includes viruses, worms, Trojan horses, and logic bombs. And beyond the external attacks, there is the issue of internal problems due to malfeasance or simple ignorance.

Security audits always begin with a risk assessment, and that is what we are describing here. First you need to identify your assets. Clearly, the actual computers, routers, switches and other devices that make up your network are assets. But it is more likely that your most important assets lie in the information on your network. Identifying assets begins with evaluating the information your network stores and its value. Does your network contain personal information for bank accounts? Perhaps medical information, health care records? In other cases, your network might contain intellectual property, trade secrets, or even classified data. Once you have identified the assets, you need to take inventory of the threats to your assets. Certainly, any threat is possible, but some are more likely than others. This is very much like what one does when selecting home insurance. If you live in a flood plain, then flood insurance is critical. If you live at a high altitude in a desert, it may be less critical. We do the same thing with our data. If you are working for a defense contractor, then foreign state-sponsored hackers are a significant threat. However, if you are the network administrator for a school district, then your greatest threat involves juveniles attempting to breach the network. It is always important to realize what the threats are for your network.

Now that you have identified your assets and inventoried the threats, you need to find out what vulnerabilities your system has. Every system has vulnerabilities. Identifying your network's specific vulnerabilities is a major part of risk assessment. The knowledge of your assets, threats, and vulnerabilities will give you the information needed to decide what security measures are appropriate for your network. You will always have budget constraints, so you will need to make wise decisions on selecting security controls. Using good risk assessment is how you make wise security decisions. issues until an incident occurs—the proverbial "closing the barn door after the horse has already gotten out." If you are fortunate, the incident will have only minor impact on your organization and will serve as a much-needed wakeup call. If you are unfortunate, then your organization may face serious and possible catastrophic consequences.

The reality is that many people who call themselves hackers are less knowledgeable than they think they are. These people have a low probability of being able to compromise any system that has implemented even moderate security precautions. This does not mean that skillful hackers do not exist, of course. However, they must balance the costs (financial, time) against the

rewards (ideological, monetary). “Good” hackers tend to target systems that yield the highest rewards. If a hacker doesn’t perceive your system as beneficial to these goals, he is less likely to expend the resources to compromise your system. It is also important to understand that real intrusions into a network take time and effort. Hacking is not the dramatic process you see in movies. It is certainly true that there are people who understand computer systems and the skills to compromise the security of many, if not most, systems. Several people who call themselves hackers, though, are not as skilled as they claim to be. They have ascertained a few buzzwords from the Internet and may be convinced of their own digital supremacy, but they are not able to affect any real compromises to even a moderately secure system.

Threats, Vulnerability, Attacks

When discussing network security, the three common terms used are as follows:

■ **Vulnerability**—A weakness that is inherent in every network and device. This includes routers, switches, desktops, servers, and even security devices themselves.

■ **Threats**—The people eager, willing, and qualified to take advantage of each security weakness, and they continually search for new exploits and weaknesses.

■ **Attacks**—The threats use a variety of tools, scripts, and programs to launch attacks against networks and network devices. Typically, the network devices under attack are the endpoints, such as servers and desktops.

Vulnerabilities

Vulnerabilities in network security can be summed up as the “soft spots” that are present in every network. The vulnerabilities are present in the network and individual devices that make up the network. Networks are typically plagued by one or all of three primary vulnerabilities or weaknesses:

■ Technology weaknesses

■ Configuration weaknesses

■ Security policy weaknesses

The sections that follow examine each of these weaknesses in more detail. Technological Weaknesses Computer and network technologies have intrinsic security weaknesses. These include TCP/IP protocol weaknesses, operating system weaknesses, and network equipment weaknesses.

Network Security Weaknesses

TCP/IP protocol weaknesses: HTTP, FTP, and ICMP are inherently insecure. Simple Network Management Protocol (SNMP), Simple Mail Transfer Protocol (SMTP), and SYN floods are related to the inherently insecure structure upon which TCP was designed.

Operating system weaknesses: The UNIX, Linux, Macintosh, Windows NT, 9x, 2K, XP, and OS/2 operating systems all have security problems that must be addressed. These are documented in the CERT archives at http://www.cert.org.

Network equipment weaknesses: Various types of network equipment, such as routers, firewalls, and switches, have security weaknesses that must be recognized and protected against. These weaknesses include the following: Password protection Lack of authentication Routing Protocols Firewall holes

Configuration Weaknesses

Network administrators or network engineers need to learn what the configuration weaknesses are and correctly configure their computing and network devices to compensate.

Unsecured user accounts: User account information might be transmitted insecurely across the network, exposing usernames and passwords to snoopers.

System accounts with easily guessed passwords: This common problem is the result of poorly selected and easily guessed user passwords.

Misconfigured Internet services: A common problem is to turn on JavaScript in web browsers, enabling attacks by way of hostile JavaScript when accessing untrusted sites. IIS, Apache, FTP, and Terminal Services also pose problems.

Unsecured default settings within products: Many products have default settings that enable security holes.

Misconfigured network equipment: Misconfigurations of the equipment itself can security problems. For example, misconfigured access lists, routing protocols, or SNMP community strings can open up large security holes. Misconfigured or lack of encryption and remote-access controls can also cause significant security issues, as can the practice of leaving ports open on a switch (which could allow the introduction of noncompany computing equipment).

Security Policy Weaknesses

Security policy weaknesses can create unforeseen security threats. The network can pose security risks to the network if users do not follow the security policy.

Lack of written security policy: An unwritten policy cannot be consistently applied or enforced.

Politics: Political battles and turf wars can make it difficult to implement a consistent security policy.

Lack of continuity: Poorly chosen, easily cracked, or default passwords can allow unauthorized access to the network.

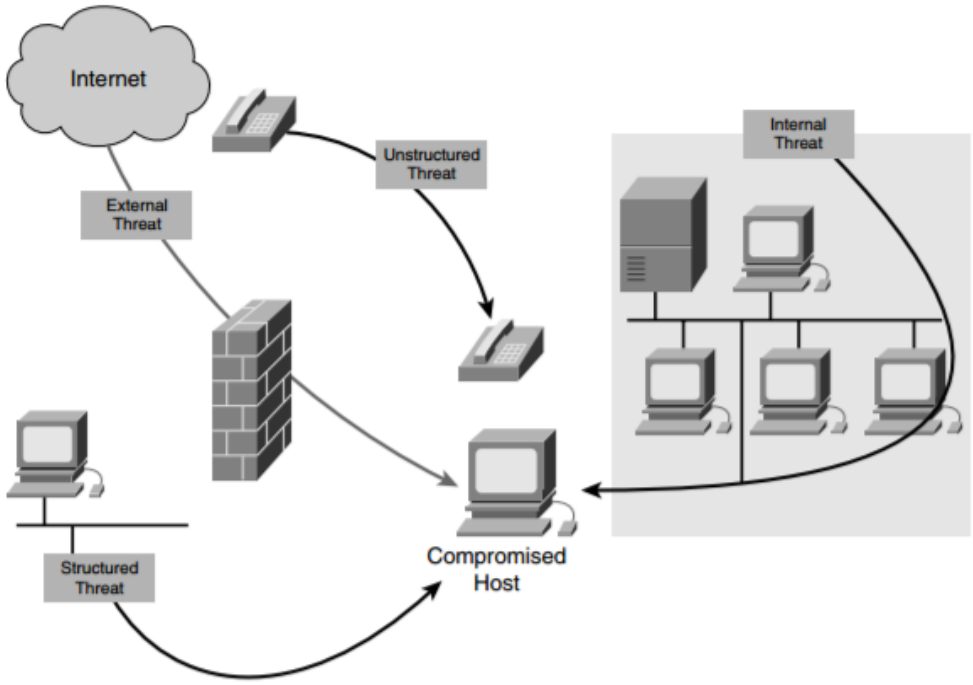
Logical access controls not applied: Inadequate monitoring and auditing allow attacks and unauthorized use to continue, wasting company resources. This could result in legal action or termination against IT technicians, IT management, or even company leadership that allows these unsafe conditions to persist. Lack of careful and controlled auditing can also make it hard to enforce policy and to stand up to legal challenges for “wrongful termination” and suits against the organization.

Software and hardware installation and changes do not follow policy: Unauthorized changes to the network topology or installation of unapproved applications create security holes.

Disaster recovery plan nonexistent: The lack of a disaster recovery plan allows chaos, panic, and is confusion to occur when someone attacks the enterprise.

Threats

There are four primary classes of threats to network security



■ **Unstructured threats**—Unstructured threats consist of mostly inexperienced individuals using easily available hacking tools such as shell scripts and password crackers. Even unstructured threats that are only executed with the intent of testing and challenging a hacker's skills can still do serious damage to a company. For example, if an external company website is hacked, the integrity of the company is damaged. Even if the external website is separate from the internal information that sits behind a protective firewall, the public does not know that. All the public knows is that the site is not a safe environment to conduct business.

■ **Structured threats**— Structured threats come from hackers who are more highly motivated and technically competent. These people know system vulnerabilities and can understand and develop exploit code and scripts. They understand, develop, and use sophisticated hacking techniques to penetrate unsuspecting businesses. These groups are often involved with the major fraud and theft cases reported to law enforcement agencies.

■ External threats—External threats can arise from individuals or organizations working outside of a company. They do not have authorized access to the computer systems or network. They work their way into a network mainly from the Internet or dialup access servers.

■ Internal threats—Internal threats occur when someone has authorized access to the network with either an account on a server or physical access to the network. According to the FBI, internal access and misuse account for 60 percent to 80 percent of reported incident

Exploits

An exploit (from the English verb to exploit, meaning "using something to one's own advantage") is a piece of software, a chunk of data, or a sequence of commands that takes advantage of a bug or vulnerability in order to cause unintended or unanticipated behavior to occur on computer software, hardware, or something electronic (usually computerized). Such behavior frequently includes things like gaining control of a computer system, allowing privilege escalation, or a denial-of-service attack.

There are several methods of classifying exploits. The most common is by how the exploit contacts the vulnerable software. A remote exploit works over a network and exploits the security vulnerability without any prior access to the vulnerable system.

A local exploit requires prior access to the vulnerable system and usually increases the privileges of the person running the exploit past those granted by the system administrator. Exploits against client applications also exist, usually consisting of modified servers that send an exploit if accessed with a client application.

Exploits against client applications may also require some interaction with the user and thus may be used in combination with the social engineering method. Another classification is by the action against the vulnerable system; unauthorized data access, arbitrary code execution, and denial of service are examples.

Many exploits are designed to provide superuser-level access to a computer system. However, it is also possible to use several exploits, first to gain low-level access, then to escalate privileges repeatedly until one reaches root.

Normally a single exploit can only take advantage of a specific software vulnerability. Often, when an exploit is published, the vulnerability is fixed through a patch and the exploit becomes obsolete until newer versions of the software become available. This is the reason why some black hat hackers do not publish their exploits but keep them private to themselves or other hackers.

Such exploits are referred to as zero day exploits and to obtain access to such exploits is the primary desire of unskilled attackers, often nicknamed script kiddies.

Pivoting

Pivoting refers to a method used by penetration testers that uses the compromised system to attack other systems on the same network to avoid restrictions such as firewall configurations, which may prohibit direct access to all machines. For example, if an attacker compromises a web server on a corporate network, the attacker can then use the compromised web server to attack other systems on the network. These types of attacks are often called multi-layered attacks. Pivoting is also known as island hopping.

Pivoting can further be distinguished into proxy pivoting and VPN pivoting. Proxy pivoting generally describes the practice of channeling traffic through a compromised target using a proxy payload on the machine and launching attacks from the computer. This type of pivoting is restricted to certain TCP and UDP ports that are supported by the proxy.

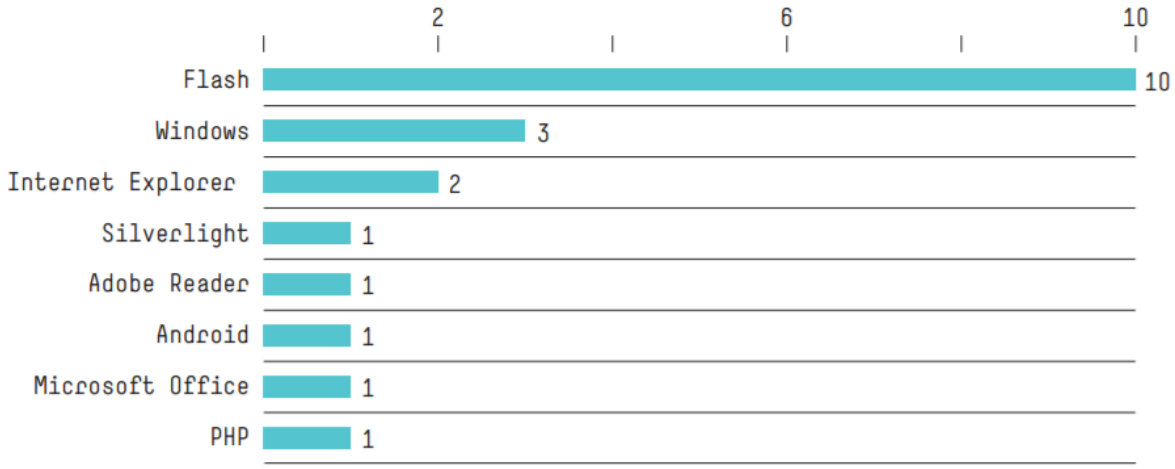
VPN pivoting enables the attacker to create an encrypted layer to tunnel into the compromised machine to route any network traffic through that target machine, for example, to run a vulnerability scan on the internal network through the compromised machine, effectively giving the attacker full network access as if they were behind the firewall.

Typically, the proxy or VPN applications enabling pivoting are executed on the target computer as the payload (software) of an exploit. Finding vulnerabilities in software is usually the domain of security researchers, with many of them participating in coordinated disclosure with vendors. Despite the progress made with a record setting year (both reporting and patching), exploits remained one of the main vectors allowing remote code execution and privilege escalation by attackers.

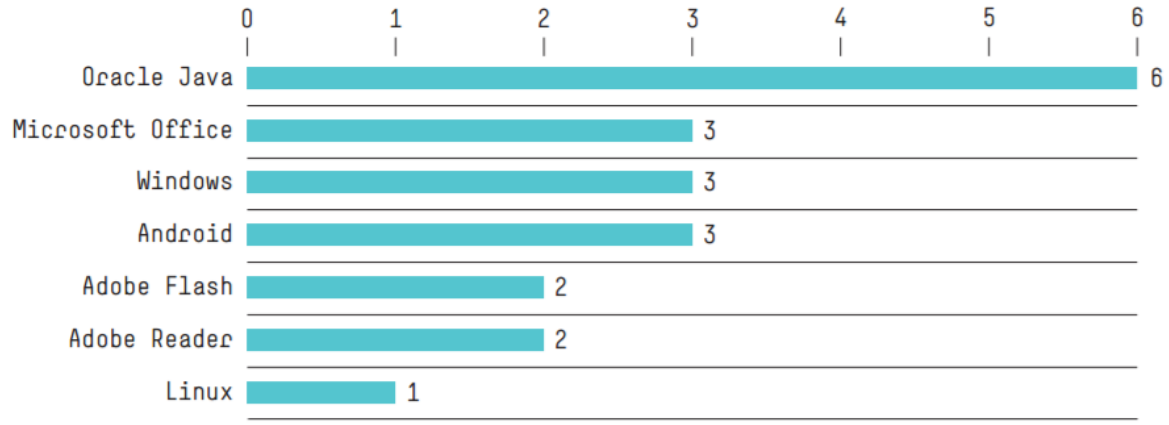
Statistics

The distribution of newly discovered samples for vulnerabilities identified in 2015 (CVE-2015-xxxx) shows the high prevalence of exploits for the Windows privilege escalation vulnerability CVE-2015-1701, which accounts for over 45% of exploit samples for the year. CVE-2015-1701, first observed in a highly-targeted attack, was used in combination with Adobe Flash remote code execution exploits.

Looking at vulnerable applications, however, the top 20 were dominated by Adobe Flash exploits. Indeed, 2015 was fraught with newly discovered Flash vulnerabilities in spite of several security improvements implemented by Adobe and Microsoft Windows such as Control Flow Guard (CFG) and a more secure Action Script vector class. Out of the top 20 applications, half affected Adobe Flash. The most commonly encountered Flash samples include two discovered after the Hacking Team breach (CVE-2015-5119 and CVE- 2015-5122), and a third (CVE-2015-0311) found in the Angler exploit toolkit.

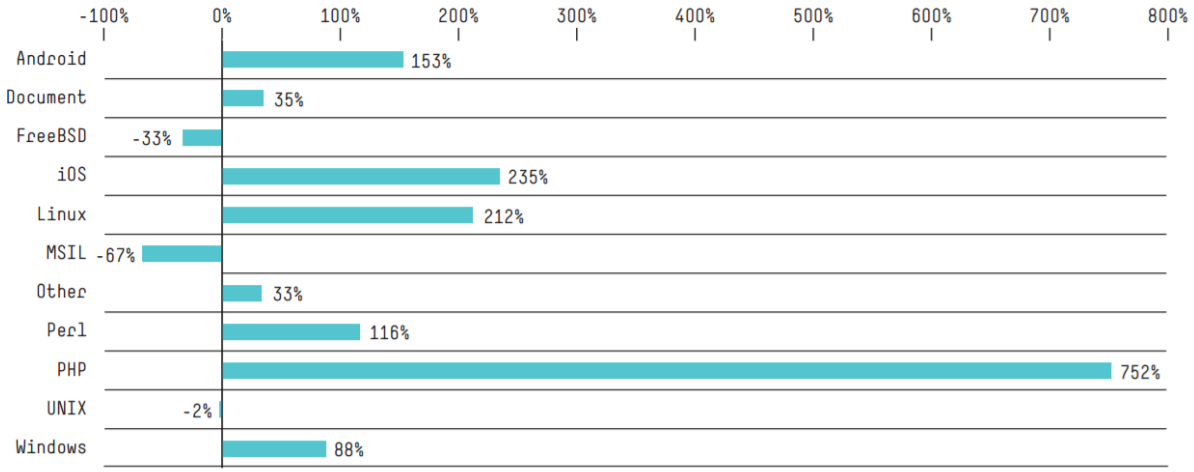


With the exception of Windows and Android, the platforms represented here are used most commonly for delivering malicious exploit files resulting in malware infections. This is a change from last year's report when Java exploits were the second most prevalent, accounting for more than 21% of all discovered exploit samples.



Regardless of increased interest in attacking mobile platforms, Microsoft Windows remains the top platform for malware with 94%. The only other platform of note here is Android, with 3% (or just over 4.5 million samples). Of interest, Android's (and other mobile platforms') main threats are potentially unwanted applications (PUAs) and advertising frameworks collecting private and potentially identifiable user information. Examining overall growth rates per platform, we see a shift away from Windows-only malware. The absolute champion, in terms of growth rate,

was Apple iOS, with an increase of more than 230% (although the number of discovered Apple malware samples, just under 70,000, is still very small compared to the number of Windows malware samples)



Malwares

Case Study

Tuesday 3:20 pm A fake but very realistic email is sent to the ten executives on the company's management team from what appears to be the CEO of a medium-sized manufacturing firm. The email is titled, "Please review this before our meeting," and it asks them to save the attachment and then rename the file extension from .zip to .exe and run the program. The program is a plug-in for the quarterly meeting happening that Friday and the plug-in is required for viewing video that will be presented. The CEO mentions in the message that the executives have to rename the attachment because the security of the mail server does not allow him to send executables. The executives do as they are told and run the program. Those who would normally be suspicious see that their fellow coworkers received the same email so it must be legitimate. Also, with the email being sent late in the day, some don't receive it until almost 5 pm and they don't have time to verify with the CEO that he sent the email. The attached file is actually a piece of malware that

installs a keystroke logger on each machine. Who would create such a thing and what would their motive be? Let's meet our attacker.

Bob Fraudster, our attacker, is a programmer at a small local company. He primarily programs using web-based technologies such as ASP.NET and supports the marketing efforts of the company by producing dynamic web pages and web applications. Bob decides that he wants to make some extra money since his job just made him take a pay cut due to the recession. Bob goes to Google.com to research bots and botnets, as he heard they can generate tons of money for operators and he thought it might be a good way to make some extra cash. Over the course of the next month or so, he joins IRC, listens to others, and learns about the various online forums where he can purchase bot software to implement click fraud and create some revenue for himself. Through Bob's research, he knows that the majority of antivirus applications can detect precompiled bots so he wants to make sure he gets a copy of source code and compiles his own bot. Bob specifically purchases a bot that communicates with his rented hosting server via SSL over HTTP, thereby reducing the chance that the outbound communications from his bots will be intercepted by security software. Since Bob is going to use SSL over HTTP, all of Bob's bot traffic will be encrypted and will go right through most content filtering technology as well. Bob signs up as an Ad Syndicator with various search engines such as Google and MSN. As an Ad Syndicator, he'll display ads from the search engine's ad rotation programs like AdSense on his website and receive a small fee (pennies) for each click on an ad that is displayed on his website. Bob uses some of the exploits he purchased with the bot in addition to some application-level vulnerabilities he purchased to compromise web servers around the world. Using standard web development tools, he modifies the HTML or PHP pages on the sites to load his ad syndication username and password so his ads are displayed instead of their own. Essentially, Bob has forced each website he has hacked into to syndicate and display ads that, when a user clicks them, will send money to him instead of the real website operators. This method of receiving money when a user clicks an advertisement on your website is called pay-per-click (PPC) advertising, and it is the root of all of Google's revenue. Next, Bob packages up the malware using the armadillo packer so it looks like a new PowerPoint presentation from the company's CEO. He crafts a specific and custom email

message that convinces the executives the attachment is legitimate and from the CEO. Now they just have to open it. Bob sends a copy of this presentation, which actually installs his bot, every 30 minutes or so to a variety of small businesses' email addresses he purchased. Since Bob had worked in marketing and implemented some email campaigns, he knows that he can purchase a list of email addresses rather easily from a company on the Internet. It is amazing how many email addresses are available for purchase on the Internet. Bob focuses his efforts on email addresses that look like they are for smaller businesses instead of corporate email addresses because he knows many enterprises use antivirus at their email gateways and he doesn't want to tip off any antivirus vendors about his bot.

Bob is smart and knows that many bots that communicate via IRC are becoming easier to detect so he purchases a bot that communicates with this privately rented server via SSL over HTTP. Using custom GET requests, the bot interacts by sending command and control messages with specific data to his web server, just like a normal browser interacts with any other website. Bob's bot communicates via HTTP so he doesn't have to worry about a firewall running on the machines he wants to infect, preventing his bot from accessing his rented web server since most firewalls allow outgoing traffic on port 443. Also, web content filtering isn't a worry for him since he is transferring data that looks innocent. Plus, when he wants to steal financial data from victims that watch the corporate PowerPoint presentation, he can just encrypt it and the web filtering will never see the data. Since he didn't release his bot using a mass propagation worm, the victim's antivirus won't detect it was installed either, as the anti-virus programs have no signatures for this bot.

Once installed, the bot runs instead of Internet Explorer as a Browser Helper Object (BHO), which gives the bot access to all of the company's normal HTTP traffic and all of the functionality of Internet Explorer such as HTML parsing, window titles, and accessing the password fields of web pages. This is how Bob's bot will sniff the data being sent to the company's credit union and the various online banks. The bot starts to connect to Bob's master bot server and queries the server to receive its list of the compromised websites to connect to and start clicking advertisements. Once the bot receives the list of links to visit, it saves the list and waits for the victim to use Internet

Explorer normally. While the victim is browsing CNN.com to learn about the latest bank bailout, the bot goes to a site in its list of links to find an ad to click. The bot understands how the ad networks work so it uses the referrer of the site the victim is actually viewing (e.g., CNN.com) to make the click on the ad look legitimate. This fools the advertisement company's antifraud software. Once the bot clicks the ad and views the advertisement's landing page, it goes off to the next link in its list. The method the bot uses makes the logs in the advertising companies' servers look like a normal person viewed the advertisement, which reduces the potential that Bob's advertising account will be flagged as fraudulent and he will be caught. In order to remain hidden and generate as much revenue for himself as possible, Bob set the bot to continue clicking advertisements in a very slow manner over the course of a couple weeks. This helps ensure the victims don't notice the extra load on their computers and that Bob's bot isn't caught for fraud. Bob has successfully converted the company's workstations into the equivalent of an ATM, spitting out cash into a street while he holds a bag to catch the money. Other stealth techniques Bob employs make sure that the search engines his hosted bot server uses to find real data don't detect his fraud either. To prevent detection, the bot uses a variety of search engines such as Google, Yahoo, AskJeeves, and so on, to implement its fraud. The more search engines it uses within the fraud scheme the more money Bob can make.

Bob needs to use the search engines because they are the conduit for the fraud. The ads clicked are from the advertisements placed on hacked websites that Bob broke into a few weeks ago. Of the ads the bot clicked on the compromised websites, only 10 percent are from Google and the rest are from other sources including other search engines. The bot implements a random click algorithm that clicks the ad link only half of the time just to make it even more undetectable by the search engine company.

Using the low and slow approach doesn't mean it will take long for Bob to start making money. For example, using just Google, let's assume Bob's stealth propagation (e.g., slowly spreads) malware infects 10,000 machines; each machine clicks a maximum of 20 ads and picks Google ads only 50 percent of the time for a total of 100,000 ads clicked. Let's also assume that

Bob chooses to display ads that when clicked will generate revenue of $0.50 per click. Using this approach, the attacker generates $50,000 in revenue (10,000 × 20 × 50% × $.50). Not bad for a couple weeks worth of work. Now that we understand Bob's motives and how he plans to attack, let's return to our factitious company and analyze how they are handling the malware outbreak. Since Bob wants to remain inconspicuous, the malware, once running, reports to a central server via SSL over HTTP and requests and sends copies of all username and passwords typed into websites by the company's employees. Because Bob built his bot using a BHO, he'll capture passwords for sites whether or not they are SSL-encrypted. Websites including the employee credit union and online e-commerce vendors such as eBay and Amazon.com are logged and sent to Bob's rented server. Since the communication is happening over SSL via HTTP to Bob's rented website, which is not flagged as a bad site by the company's proxy, nothing is blocked.

Wednesday 8:00 am The malware propagates by sending itself to all the users in the corporate address book of the executives who received the same message from the CEO. It also starts infecting other machines by exploiting network vulnerabilities in the unpatched machines and machines that are running older versions of Microsoft Windows that IT hasn't had a chance to update yet. Why didn't the CIO approve the patch management product the network security team proposed to buy and implement last year?

Wednesday 4:00 pm Hundreds of employees are now infected, but the rumor of the application from the email needing to be installed has reached IT, and they start to investigate. IT finds that this may be malware, but their corporate antivirus and email antivirus didn't detect it so they aren't sure what the executable does. They have no information about the executable being malicious, its intent, or how the malware operates. They place their trust in their security vendors and send samples to their antivirus vendor for analysis.

Thursday 10:00 am IT is scrambling and attempting to remove the virus using the special signatures received from the antivirus vendor last night. It is a cat-and-mouse game with IT barely keeping

ahead of the propagation. IT decided to turn off all workstations companywide last night, including those that were required by the manufacturing firm's order processors in London. Customers were not happy.

Thursday 8:00 pm IT is still attempting to disinfect the workstations. An IT staff member starts to do analysis on his own and discovers the binary may have been written by an ex-employee based off of some strings located in the binary that reference a past scuffle between the previous CIO and Director of IT. IT contacts the FBI to determine if this could be a criminal act.

Friday 9:00 am The quarterly meeting is supposed to start but is delayed because the workstation that the CEO must use to give his presentation was infected and hasn't been cleaned since the machine was off when IT pushed out the new antivirus updates. The CEO calls an emergency meeting with the CIO to determine what is happening. IT continues to disinfect the network and is making steady progress.

Saturday 11:00 am IT feels that they have completely removed the malware from the network. Employees will be ready to work on Monday, but IT will still have much to do as the infection caused so much damage that 30 workstations have to be rebuilt because the malware was not perfectly removed from each workstation.

Next Monday 3:00 pm The CIO meets with the CEO to give an estimated cost to the time spent in cleaning up the problem. Neither the CEO nor the CIO is able to fathom the actual number of lost sales or productivity of the 1500 workers who were infected and not able to work. Furthermore, the CIO informs the CEO that a few employees had their identities stolen since the malware logged their keystrokes as they logged into their online bank account. The victim employees want to know what the company is going to do to help them.

Situations like the above are not uncommon. The technical details may be different for each case but the meeting on Monday that the CIO had with the CEO is all too common. No one

within the manufacturing organization anticipated this it seemed, yet the industry trade magazines and every security report has said this was inevitable. The main issue in this case is that the company was unprepared. As in war, knowledge is half the battle, and yet most organizations do not understand malware, how it is written, and why it is written, and they don't have adequate policies and processes in place to handle a full-scale bot outbreak. Because of this, in 2008, the second highest cost to an organization from malware was the cost to remove bots from the network. In our case study, the total time IT had to dedicate to get the
business back up and running was high and that amount does not include any potential notifications, compliance violations, or legal costs that are the result of the malware capturing personally identifiable information.

Introduction

Today's threat landscape is more hostile than ever before. Recent advances in phishing and spam have shown that the attacker's methods have become more psychological than technological. Users are now targeted via email and the Web and asked to give up their sensitive information, such as usernames and passwords for online banking, by websites that look so credible many people cannot even tell the difference. According to McAfee's Site Advisor, 95 percent of over 120 thousand people who have taken their Spyware Quiz, a test that asks whether a site is safe or not, incorrectly assume a site is safe when it is verified to contain malware. McAfee's quiz is a stunning example of the problem users face. They must decide whether something will negatively affect their machine with a quick visual inspection. Given the lack of security awareness, this important decision is akin to a four-year-old boy trying to determine if his dad really did pull a quarter from his ear or not. Once the attacker has fooled the user into downloading the malware, the attacker is free to explore the newest frontier in cyberspace—your workstation—for confidential information, usernames, and passwords, and personally identifiable information such as your Social Security Number or bank account information. When was the last time you heard about a major virus outbreak on your local news? Two years ago?

Viruses are dead. The threat of worms and viruses to home users and corporate networks has dropped dramatically since the major outbreaks of Bagle and Netsky in 2004. However, the

outbreaks did not stop because virus writers decided to pack up and go home. Instead, they stopped because their main goal, publicity, was no longer interesting. They wanted something more, such as money, sensitive information, and sustained access to unauthorized systems, to leverage those system resources, so they changed their methods, techniques, and tools, aligning them with their new motives to be discreet and target-focused. Thus began the era of malware and rootkits. Some of the changes malware authors have experienced were forced upon them as the security industry elevated the security arms race to new levels. A decrease in the number of unauthenticated remote vulnerabilities within Microsoft's operating system and the increased usage of perimeter security products forced attackers to elevate their game to a new level.

Malware

Malware, short for malicious software, is any software used to disrupt computer or mobile operations, gather sensitive information, gain access to private computer systems, or display unwanted advertising. Before the term malware was coined by Yisrael Radai in 1990, malicious software was referred to as computer viruses. The first category of malware propagation concerns parasitic software fragments that attach themselves to some existing executable content. The fragment may be machine code that infects some existing application, utility, or system program, or even the code used to boot a computer system. Malware is defined by its malicious intent, acting against the requirements of the computer user, and does not include software that causes unintentional harm due to some deficiency.

Malware may be stealthy, intended to steal information or spy on computer users for an extended period without their knowledge, as for example Regin, or it may be designed to cause harm, often as sabotage (e.g., Stuxnet), or to extort payment (CryptoLocker). 'Malware' is an umbrella term used to refer to a variety of forms of hostile or intrusive software, including computer viruses, worms, trojan horses, ransomware, spyware, adware, scareware, and other malicious programs. It can take the form of executable code, scripts, active content, and other software. Malware is often disguised as, or embedded in, non-malicious files. As of 2011 the majority of active malware threats were worms or trojans rather than viruses. In law, malware is sometimes known as a computer contaminant, as in the legal codes of several U.S. states.

Spyware or other malware is sometimes found embedded in programs supplied officially by companies, e.g., downloadable from websites, that appear useful or attractive, but may have, for example, additional hidden tracking functionality that gathers marketing statistics. An example of such software, which was described as illegitimate, is the Sony rootkit, a Trojan embedded into CDs sold by Sony, which silently installed and concealed itself on purchasers' computers with the intention of preventing illicit copying; it also reported on users' listening habits, and unintentionally created vulnerabilities that were exploited by unrelated malware. Software such as anti-virus and firewalls are used to protect against activity identified as malicious, and to recover from attacks.

Why They Want Your Workstation

Technology advances and the availability of attack vectors were factors in attackers changing their methods, but their target, you, ultimately made the decision for them. Authors of malware and rootkits realized that they could generate revenue for themselves by utilizing the malware they were creating to steal sensitive data, such as your online banking username and password, commit click fraud, and sell remote control of infected workstations to spammers as spam relays. They could actually receive a return on investment from the time they put into writing their malware. Your workstation was now worth much more than it was before; therefore, the attacker’s tools needed to adapt to maintain control of the infected workstation as well as infect as many workstations as possible. The home user is not the only target of malware authors. The corporate workstation is just as juicy and inviting. Enterprise workstation users routinely save confidential corporate documents to their local workstation, log into personal accounts online such as bank accounts, and log into corporate servers that contain corporate intellectual property.

All of these items are of interest to attackers and are routinely gathered during malware infections. A very recent example of an “enterprise” target is U.S. Presidential Candidates Barack Obama and John McCain. Both candidates’ campaign systems were attacked and infiltrated by remote attackers. We can only guess at the type of information they were looking for, but the data

they had access to, if it was released, could have caused significant damage to either campaign. Even what may seem like useless information is routinely stolen and sold or distributed. Items such as personal photos, secret love affair chats, which may also occur at the workplace, and email are targets as well.

Intent Is Hard To Detect

The change in landscape has increased the technical challenges for malware authors, but the greatest change has been a change in intent. As mentioned before, many virus authors were writing viruses purely for ego gratification and to show off to their friends. Virus writers were part of an underground subculture that rewarded members for new techniques and for mass destruction. The race to be the smartest author caused many virus authors to push the envelope and actually release their creations, causing massive amounts of damage. These acts are synonymous with the plot of many bad movies where two boys constantly try to "one up" each other when fighting over a girl in high school but all they leave is destruction in their wake. In the end, neither gets the girl and the two boys end up in trouble and looking stupid. The same is true for virus authors who released viruses. In countries where writing viruses is illegal, the virus writers were caught and prosecuted.

Some virus authors weren't in it for ego but for protest, as was the case with Onel A. De Guzman. De Guzman was seen as a Robinhood in the Philippines. He wrote the portion of the ILOVEYOU virus that stole the usernames and passwords people used to access the Internet and gave the information to others to utilize. In the Philippines, where Internet access costs as much as $90 per month, many saw his virus as a great benefit. In addition to de Guzman, Dark Avenger, a Bulgarian virus author, was cited as saying he wrote viruses and released them "because they gave him a sense of political power and freedom he was denied in Bulgaria." Malware and rootkits are not about ego or protest— they're about money.

Malware authors want money, and the easiest way to get it is to steal it from you. Their intent with the programs they have written has changed dramatically. Malware and rootkits are now precision-theft tools, not billboards for shouting their accolades and propaganda to friends.

Why does this shift matter? The shift to malicious intent by authors sent a signal to those who protect users from malware that they needed to shift their detection and prevention capabilities. Viruses and worms are technical anomalies. In general, their functionality is not composed of a common set of features that normal computer users may execute, such as a word processing application; therefore, detecting and preventing an anomaly is easier than detecting a user doing something malicious. The problem with detecting malicious intent is in who defines what is malicious. Is it the antivirus companies or the media? Different computer users have different risk tolerances so one person may be able to tolerate a piece of malware running in return for the benefit it may provide (we will get to the benefits malware delivers later), whereas someone else may not tolerate any malware. Understanding the intent of a legitimate user's action is hard, if not impossible. Governments around the world have been trying to understand the intent of human action within the law enforcement and legal system for years with little success.

Conviction rates in most countries following an Anglo-Saxon legal system (such as the United States) range from 40 to 80 percent. If the legal systems around the world, which have been dealing with this problem for hundreds of years, have a hard time determining intent, how do we stand a chance in stopping malware? We believe we do, but the battle is one that we have never seen before in the cyberwarfare community, which is why the remainder of the booklet focuses on arming you with the technical knowledge about how malware propagates, infects, maintains control, and steals data. Hopefully, armed with this information, you will be able to determine the intent of the applications running on your workstation and take the first step in defending your network against malware.

It's A Business

As mentioned previously, malware authors are focused on making a profit. Like all entrepreneurs who want to make money, they start various businesses to take advantage of the situation. The largest and most active of all the malware groups is the Russian Business Network (RBN). Russia has been on the malware scene for years, with many of the most well-known viruses and Trojans, such as Bagle, MyDoom, and Netsky, originating from Russian developers. It seems that because of the lack of high-paying IT jobs within Russia and the fact that the majority of the

IT jobs are mundane and very task-oriented, the large base of young professionals with high levels of technical talent are turning to crime to get their technology fix. Before we dive into the business of the RBN, let's explore the organization. The RBN is nothing more than a highly scalable, redundant, and efficient hosting platform that just happens to host malware. Its hosting customers include child pornography sites, gambling, malware, and phishing sites. The RBN doesn't care what the hosting platform is used for as long as it receives revenue.

The RBN primarily focuses its efforts into six areas:

- Phishing
- Malware
- Scams
- Distributed denial of service (DDoS)
- Pornography (including child pornography)
- Games

In order to support these efforts, the RBN has created and deployed a hosting platform that consists of one main requirement—bandwidth—and continually deploys malicious webservers, botnets, and command and control servers. The RBN began to be seen as a distributor of malware in 2005 when it was discovered that the CoolWebSearch Malware was being distributed by servers hosted on RBN address space. The RBN continued to increase their distribution and hosting of malware through the use of exploits such as the Microsoft VRML exploit in 2006. The RBN used a variety of exploits and malware during anonymous customer attacks but its footprint was still relatively small. Starting in 2007, with the release of the MPack attack toolkit, the RBN started to really take hold of the malware market. Although MPack may not have actually been written by the RBN, the author of MPack is Russian, and many of the initial MPack installations, including Torpig, a known malware payload, have been traced back to the RBN network. MPack was sold to attackers for $500 to $1000 and an extra $300 included a loader to help jumpstart the malicious activities. MPack was a great step forward for the RBN as it contained over ten different exploits and attackers could choose which exploit to use based on the

connecting target. It was very effective and gave the RBN something they had never really had before: metrics. Since MPack contained multiple exploits, the management console detailed which web browsers were most successfully infected, what country the web browsers originated from, and infection ratios. These metrics allowed attackers to finetune their attacks or sell a specific type of infected machine based on their inventory. Continuing the infection spree, the RBN appears to have been behind the Bank of India incident in which the website for the Bank of India began distributing malware from the RBN's network. Amazingly, the Bank of India site attempted to install over 20 different types of malware on a client's computer. RBN was now definitely in the volume game of malware distribution!.

Malware distribution is the RBN's number one activity, but phishing is a close second. The amount of disinformation and incorrect information available about the RBN has made it very difficult to link the network directly to a specific phishing attack; however, significant data shows that the RBN networks have hosted banking Trojans and other services that enabled updates to bypass antivirus, phishing content pages, and have acted as a destination for logs from installed Trojans.

The RBN, like any entrepreneurial business, has also launched retail sites that accept credit cards for fake anti-malware software and has entered into partnerships with traditional hackers in order to increase its footprint of web servers that are serving malicious traffic. With the RBN's massive organization and infrastructure, it is easy to see that the estimated revenue for all the RBN's activities is around $120 million per year. With that type of revenue, you can see why the goal of the attackers has moved from owning the server to owning identities.

Elk Cloner

Elk Cloner is one of the first known microcomputer viruses that spread "in the wild", i.e., outside the computer system or laboratory in which it was written. It attached itself to the Apple II operating system and spread by floppy disk. It was written around 1982 by programmer and entrepreneur Rich Skrenta as a 15-year-old high school student, originally as a joke, and put onto a game disk.

Elk Cloner spread by infecting the Apple DOS 3.3 operating system using a technique now known as a boot sector virus. It was attached to a game; the game was then set to play. The 50th time the game was started, the virus was released, but instead of playing the game, it would change to a blank screen that displayed a poem about the virus named Elk Cloner. If a computer booted from an infected floppy disk, a copy of the virus was placed in the computer's memory. When an uninfected disk was inserted into the computer, the entire DOS (including Elk Cloner) would be copied to the disk, allowing it to spread from disk to disk. To prevent the DOS from being continually re-written each time the disk was accessed, Elk Cloner also wrote a signature byte to the disk's directory, indicating that it had already been infected.

The poem that Elk Cloner would display was as follows:

```
Elk Cloner:  The program with a personality

   It will get on all your disks
     It will infiltrate your chips
       Yes it's Cloner!

   It will stick to you like glue
     It will modify ram too
       Send in the Cloner!
```

Elk Cloner did not cause deliberate harm, but Apple DOS disks without a standard image had their reserved tracks overwritten. Elk Cloner was created by Skrenta as a prank in 1982. Skrenta already had a reputation for pranks among his friends because, in sharing computer games and software, he would often alter the floppy disks to shut down or display taunting on-screen messages. Due to this reputation, many of his friends simply stopped accepting floppy disks from him. Skrenta thought of methods to alter floppy disks without physically touching or harming them. During a winter break from Mt. Lebanon High School in Mt. Lebanon, Pennsylvania, Skrenta discovered how

to launch the messages automatically on his Apple II computer. He developed what is now known as a boot sector virus, and began circulating it in early 1982 among high school friends and a local computer club. Twenty-five years later in 2007, Skrenta called it "some dumb little practical joke." According to contemporary reports, the virus was rather contagious, successfully infecting the floppies of most people Skrenta knew, and upsetting many of them.

Part of the "success", of course, was that people were not at all wary of the potential problem, nor were virus scanners or cleaners available. The virus could be removed using Apple's MASTER CREATE utility or other utilities to re-write a fresh copy of DOS to the infected disk. Furthermore, once Elk Cloner was removed, the previously-infected disk would not be re-infected since it already contained the Elk Cloner "signature" in its directory. It was also possible to "inoculate" uninfected disks against Elk Cloner by writing the "signature" to the disk; the virus would then think the disk was already infected and refrain from writing itself.

A Brief History Of Malware

In January 2011, Brain - the first PC-based malware - turned 25 years old. (It's worth noting that the first computer virus was actually a Mac virus, Elk Cloner, in 1982). Following is a brief history of the first 25 years of malware evolution.

In 1986, most viruses were found in universities and propagation was primarily via infected floppy disks. Notable malware included Brain (1986), Lehigh, Stoned, and Jerusalem (1987), the Morris worm (1988), and Michelangelo - the first headline grabber - in 1991.

By the mid-90s, businesses were equally impacted (in large part due to macro viruses) and propagation had moved to the network. Notable malware for the period included DMV - the first proof of concept macro virus - in 1994, Cap.A - the first high risk macro virus - in 1997, CIH (aka Chernobyl) - the first virus to damage hardware - in 1998.

By the latter part of the 90s, viruses had begun impacting home users as well and email propagation was ramping up. Notable malware included Melissa (the first widespread email worm) and Kak - the first and one of the very few true email viruses - both in 1999.

At the start of the new millennium, Internet and email worms were making headlines across the globe. Notables included Loveletter - the first high-profile profit-motivate malware (May 2000), the Anna Kournikova email worm (Feb 2001), the March 2001 Magistr (which, like CIH before it, also impacted hardware), the Sircam email worm in July 2001 which harvested files from the My Documents folder, the CodeRed Internet worm in August 2001, and Nimda - a Web, email and network worm - in September 2001.

As the decade progressed, malware almost exclusively became a profit motivated tool. Throughout 2002 and 2003, Web surfers were plagued by out-of-control popups and other Javascript bombs. FriendGreetings ushered in manually driven socially engineered worms in October 2002 and SoBig began surreptitiously installing spam proxies on victim computers.

Phishing and other credit card scams also took off during the period. Other notable threats for the period included the Blaster and Slammer Internet worms. In January 2004, an email worm war broke out between the authors of MyDoom, Bagle and Netsky. Ironically, this led to improved email scanning and higher adoption rates of email filtering, which eventually spelled a near demise of mass-spreading email worms.

The November 2005 discovery and disclosure of the now infamous Sony rootkit led to the eventual inclusion of rootkits in most modern day malware. Pump & Dump and money mule job scams joined the growing numbers of Nigerian 419 scams, phishing, and lottery scams in 2006. Though not directly malware-related, such scams were a continuation of the theme of profit-motivated criminal activity launched via the Internet.

Website compromises escalated in 2007 due in large part to the discovery and disclosure of MPack, a crimeware kit used to deliver exploits via the Web. Notable compromises included the Miami Dolphins stadium site, Tomshardware.com, TheSun, MySpace, Bebo, Photobucket and The India Times websites.

By the end of 2007, SQL injection attacks had begun to ramp up, netting victim sites such as the popular cuteoverload.com and Ikea websites. By January 2008, Web attackers were employing stolen FTP credentials and leveraging weak configurations to inject iframes on tens of thousands

of mom & pop style websites, the so-called long tail of the Web. In June 2008, the Asprox botnet facilitated automated SQL injection attacks, claiming walmart.com as one of its victims.

Advanced persistent threats emerged during this same period as attackers began segregating victim computers and delivering custom configuration files to those of highest interest. In early 2009, Gumblar - the first dual botnet - emerged. Gumblar not only dropped a backdoor on infected PCs and used it to steal FTP credentials, it used those credentials to hide a backdoor on compromised websites as well. This development was quickly adopted by other Web attackers. The result: today's website compromises no longer track back to a handful of malicious domain hosts - instead any of the thousands of compromised sites can interchangeably play the role of malware host.

The volume of malware is merely a by-product of distribution and purpose. This can best be seen by tracking the number of known samples based on the era in which it occurred. For example, during the late 80s most malware were simple boot sector and file infectors spread via floppy disk. With limited distribution and less focused purpose, unique malware samples recorded in 1990 by AV-Test.org numbered just 9,044.

As computer network adoption and expansion continued through the first half of the 90s, distribution of malware became easier and malware volume increasd. In 1994, AV-Test.org reported 28,613 unique malware samples (based on MD5). As technologies standardized, certain types of malware were able to gain ground. Macro viruses which exploited Microsoft Office products not only achieved greater distribution via email, they also gained a distribution boost by the increased adoption of email. In 1999, AV-Test.org recorded 98,428 unique malware samples.

As broadband Internet adoption increased, Internet worms became more viable. Distribution was further accelerated by increased use of the Web and the adoption of so-called Web 2.0 technologies which fostered a more favorable malware environment. In 2005, AV-Test.org recorded 333,425 unique malware samples. Increased awareness in Web-based exploit kits led to an explosion of Web-delivered malware throughout the latter part of the millennium's first decade.

In 2006, the year MPack was discovered, AV-Test.org recorded 972,606 unique malware samples. As automated SQL injection and other forms of mass website compromises increased distribution capabilities in 2007, malware volume made its most dramatic jump, with 5,490,960 unique samples recorded by AV-Test.org in that year. Since 2007, the number of unique malware has continued exponential growth, doubling or more each year since. Currently, vendors estimates of new malware samples range from 30k to over 50k per day. Put another way, the current monthly volume of new malware samples is greater than the total volume of all malware from 2006 and previous years.

Antivirus / Security Revenue

During the "sneakernet" era in the late 80s and early 90s, antivirus vendor revenues were collectively less than $1B USD. By 2000, antivirus revenues had increased to ~ $1.5B.

While some may point to the increasing antivirus/security vendor revenues as "proof" that antivirus vendors profit from (and thus create) malware, the math itself does not bear out this conspiracy theory. In 2007, for example, antivirus revenues grew by 131% - but malware volumes increased 565% that year. Additionally, antivirus revenue increases are also the result of new companies and expanding technologies (for example, security appliances and cloud-based security developments).

Melissa virus

Melissa is a fast-spreading macro virus that is distributed as an e-mail attachment that, when opened, disables a number of safeguards in Word 97 or Word 2000, and, if the user has the Microsoft Outlook e-mail program, causes the virus to be resent to the first 50 people in each of the user's address books. While it does not destroy files or other resources, Melissa has the potential to disable corporate and other mail servers as the ripple of e-mail distribution becomes a much larger wave. On Friday, March 26, 1999, Melissa caused the Microsoft Corporation to shut down incoming e-mail. Intel and other companies also reported being affected. The U. S. Department of Defense-funded Computer Emergency Response Team (CERT) issued a warning about the virus and developed a fix.

How Melissa Works

Melissa arrives in an attachment to an e-mail note with the subject line "Important Message from [the name of someone]," and body text that reads "Here is that document you asked for...don't show anyone else ;-)". The attachment is often named LIST.DOC. If the recipient clicks on or otherwise opens the attachment, the infecting file is read to computer storage. The file itself originated in an Internet alt.sex newsgroup and contains a list of passwords for various Web sites that require memberships. The file also contains a Visual Basic script that copies the virus-infected file into the normal.dot template file used by Word for custom settings and default macros. It also creates this entry in the Windows registry:

```
HKEY_CURRENT_USERSoftwareMicrosoftOffice"Melissa?"="...by Kwyjibo"
```

The virus then creates an Outlook object using the Visual Basic code, reads the first 50 names in each Outlook Global Address Book, and sends each the same e-mail note with virus attachment that caused this particular infection. The virus only works with Outlook, not Outlook Express.

In a small percentage of cases (when the day of the month equals the minute value), a payload of text is written at the current cursor position that says:

"Twenty-two points, plus triple-word score, plus fifty points for using all my letters. Game's over. I'm outta here."

The quote refers to the game of Scrabble and is taken from a Bart Simpson cartoon. The virus also disables some security safeguards. These are described by CERT and the anti-virus software sites.

Computer Virus

A computer virus is a small program written to alter the way a computer operates, without the permission or knowledge of the user. A virus must meet two criteria:

- It must execute itself. It will often place its own code in the path of execution of another program.
- It must replicate itself. For example, it may replace other executable files with a copy of the virus infected file. Viruses can infect desktop computers and network servers alike.

Some viruses are programmed to damage the computer by damaging programs, deleting files, or reformatting the hard disk. Others are not designed to do any damage, but simply to replicate themselves and make their presence known by presenting text, video, and audio messages. Even these benign viruses can create problems for the computer user. They typically take up computer memory used by legitimate programs. As a result, they often cause erratic behavior and can result in system crashes. In addition, many viruses are bug-ridden, and these bugs may lead to system crashes and data loss. Viruses need time to infect. Not all viruses attack, but all use system resources and often have bugs. Viruses come in a great many different forms, but they all potentially have two phases to their execution, the infection phase and the attack phase.

Infection Phase

When the virus executes it has the potential to infect other programs. What's often not clearly understood is precisely when it will infect the other programs. Some viruses infect other programs each time they are executed; other viruses infect only upon a certain trigger. This trigger could be anything; a day or time, an external event on your PC, a counter within the virus, etc. Virus writers want their programs to spread as far as possible before anyone notices them.

It is a serious mistake to execute a program a few times – find nothing infected and presume there are no viruses in the program. You can never be sure the virus simply hasn't yet triggered its infection phase. Many viruses go resident in the memory of your PC in the same or similar way as terminate and stay resident (TSR) programs. (For those not old enough to remember TSRs, they were programs that executed under DOS but stayed in memory instead of ending.) This means the virus can wait for some external event before it infects additional programs. The virus may silently lurk in memory waiting for you to access a diskette, copy a file, or execute a program, before it infects anything. This makes viruses more difficult to analyze since it's hard to guess what trigger condition they use for their infection.

On older systems, standard (640K) memory is not the only memory vulnerable to viruses. It is possible to construct a virus which will locate itself in upper memory (the space between 640K and 1M) or in the High Memory Area (the small space between 1024K and 1088K). And, under

Windows, a virus can effectively reside in any part of memory. Resident viruses frequently take over portions of the system software on the PC to hide their existence. This technique is called stealth. Polymorphic techniques also help viruses to infect yet avoid detection. Note that worms often take the opposite approach and spread as fast as possible. While this makes their detection virtually certain, it also has the effect of bringing down networks and denying access; one of the goals of many worms.

Attack Phase

Many viruses do unpleasant things such as deleting files or changing random data on your disk, simulating typos or merely slowing your PC down; some viruses do less harmful things such as playing music or creating messages or animation on your screen. Just as the infection phase can be triggered by some event, the attack phase also has its own trigger.

Does this mean a virus without an attack phase is benign? No. Many viruses have bugs in them and these bugs often cause unintended negative side effects. In addition, even if the virus is perfect, it still steals system resources.

Viruses often delay revealing their presence by launching their attack only after they have had many opportunities to spread. This means the attack could be delayed for days, weeks, months, or even years after the initial infection.

The attack phase is optional, many viruses simply reproduce and have no trigger for an attack phase. Does this mean that these are “good” viruses? No! Anything that writes itself to your disk without your permission is stealing storage and CPU cycles. This is made worse since viruses that “just infect,” with no attack phase, often damage the programs or disks they infect. This is not an intentional act of the virus, but simply a result of the fact that many viruses contain extremely poor quality code.

An example, one of the most common past viruses, Stoned, is not intentionally harmful. Unfortunately, the author did not anticipate the use of anything other than 360K floppy disks. The original virus tried to hide its own code in an area of 1.2MB diskettes that resulted in corruption of the entire diskette (this bug was fixed in later versions of the virus).

Boot Sector Virus

Boot sector is a physical sector, or section, on a hard drive that includes information about how to start the boot process in order to load an operating system. A boot sector exists on an internal hard drive where an operating system like Windows is installed, as well on storage devices that you may not even need to boot from, but instead are just holding personal data on, like an external hard drive, floppy disk, or other USB device.

How the Boot Sector is Used

Once a computer turns on, the very first thing that happens is that BIOS looks for clues on what it needs to start up the operating system. The first-place BIOS will look is the first sector of each storage device connected to computer. Say you have one hard drive in your computer. This means you have one hard drive that has one boot sector. In that section of the hard drive may be one of two things: The Master Boot Record (MBR) or the Volume Boot Record (VBR).

The MBR is very first sector of any formatted hard drive. Since BIOS looks at the first sector to understand how it should proceed, it will load the MBR in to memory. Once the MBR data is loaded, the active partition can be found so that the computer knows where the operating system is located. If a hard drive has multiple partitions, the VBR is the first sector within each partition. The VBR is also the first sector of a device that isn't partitioned at all.

Beyond running the risk of being corrupted by some sort of accident or hardware failure, the boot sector is also a common area for malware to take hold. Malware makers love focusing their attention on the boot sector because its code is launched automatically and sometimes without protection, before the operating system even starts up! A boot sector infector is a virus that inserts itself into the boot sector of a disk. Many boot sector viruses will stop your computer from starting all the way, making scanning for malware from within Windows impossible.

EXAMPLE: The Brain virus for the IBM PC is a boot sector infector. When the system boots from an infected disk, the virus is in the boot sector and is loaded. It moves the disk interrupt vector (location 13H or 19) to an alternative interrupt vector (location 6DH or 109) and sets the disk

interrupt vector location to invoke the Brain virus now in memory. It then loads the original boot sector and continues the boot. Whenever the user reads a floppy, the interrupt at location 13H is invoked. The Brain virus checks for the signature 1234H in the word at location 4. If the signature is present, control is transferred to the interrupt vector at location 6DH so that a normal read can proceed. Otherwise, the virus infects the disk. To do this, it first allocates to itself three contiguous clusters (of two contiguous sectors each). The virus then copies the original boot sector to the first of the six contiguous sectors and puts copies of itself into the boot sector and the remaining five sectors.

If there are no unused clusters, the virus will not infect the disk. If it finds only one unused cluster, it will simply overwrite the next two. System sector viruses modify the program in either the DOS boot sector or the Master Boot Record. Since there isn't much room in the system sector (only 512 bytes), these viruses usually have to hide their code somewhere else on the disk. These viruses sometimes cause problems when this spot already contains data that is then overwritten.

Some viruses, such as the Pakistani Brain virus, mark the spot where they hide their code as bad. This is one reason to be suspicious if CHKDSK suddenly reports additional bad sectors on your disk and you don't know why (don't panic, bad sectors occur frequently for a wide variety of reasons). These viruses usually go resident in memory on your PC, and infect any floppy disk that you access. Simply doing a DIR on a floppy disk may cause it to be infected if one of these viruses is active in memory.

On Macintosh systems, some viruses will even infect a diskette immediately upon inserting a diskette into the floppy drive. (PCs running under DOS do not access a disk automatically.) Since viruses are active in memory (resident), they can hide their presence. If Brain is active on your PC, and you use a sector editor to look at the boot sector of an infected diskette, the virus will intercept the attempt to read the infected boot sector and return instead a saved image of the original boot sector. You will see the normal boot sector instead of the infected version.

Removing a boot sector virus can be difficult because it may encrypt the boot sector. In many cases, users may not even be aware they have been infected with a virus until they run an antivirus

protection program or malware scan. As a result, it is critical for users to rely on continually updated virus protection programs that have a large registry of boot viruses and the data needed to safely remove them. If the virus cannot be removed due to encryption or excessive damage to existing code, the hard drive may need reformatting to eliminate the infection.

Macro Virus

In computers, a macro (for "large"; the opposite of "micro") is any programming or user interface that, when used, expands into something larger. The original use for "macro" or "macro definition" was in computer assembler language before higher-level, easier-to-code languages became more common. In assembler language, a macro definition defines how to expand a single language statement or computer instruction into a number of instructions. The macro statement contains the name of the macro definition and usually some variable parameter information. Macros were (and are) useful especially when a sequence of instructions is used a number of times (and possibly by different programmers working on a project). Some pre-compilers also use the macro concept. In general, however, in higher-level languages, any language statement is about as easy to write as an assembler macro statement.

Assembler macros generate instructions inline with the rest of a program. More elaborate sequences of instructions that are used frequently by more than one program or programmer are encoded in subroutines that can be branched to from or assembled into a program.

Macro viruses add their code to the macros associated with documents, spreadsheets, and other data files. The first macro virus, called Concept, appeared in July 1995 and macro viruses (mostly infecting Word documents) subsequently became the dominant type of virus until the turn of the century, when Microsoft disabled macros by default in Office (versions since Office 2000): since then, cybercriminals have had to try and trick their victims into enabling macros before their infected macro is able to run.

How Macro Viruses Spread

Macro viruses are most commonly found embedded in documents or inserted as malicious code into word-processing programs. They may come from documents attached to emails, or the

code may be downloaded after clicking on "phishing" links in banner ads or URLs. They are difficult to detect, as they do not operate until an infected macro is run, at which time they perform a series of commands. A macro virus is similar to a Trojan virus, since it may appear benign and users may not immediately notice any ill effects. Unlike Trojans, however, macro viruses can replicate themselves and infect other computers.

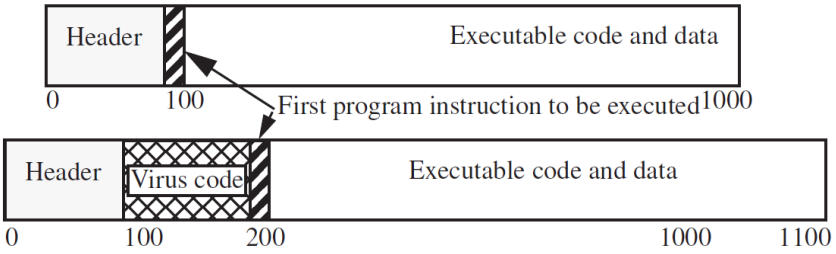
The main risk of macro viruses is their ability to spread quickly. Once an infected macro is run, all other documents on a user's computer become infected. Some of these viruses cause abnormalities in text documents, such as missing or inserted words, while others access email accounts and send out copies of infected files to all of a user's contacts, who in turn open and access these files because they come from a trusted source.

These viruses can also be designed to erase or compromise stored data. In addition, it's important to note that macro viruses are cross-platform; they can infect both Windows and Mac computers using the same code. Any program that uses macros can operate as a host, and any copy of an infected program — sent via email, stored on disk or on a USB drive — will contain the virus.

To remove these viruses, users should rely on security software that provides specific macro virus detection and removal tools. Regular scans will clean any infected documents and ensure no new computer viruses are downloaded. Macro viruses can be found in several different forms. Although some people consider them to be a relic of the late 1990s, they have in fact made a resurgence in recent years, forcing users to be extra vigilant.

Executable Infectors

The PC variety of executable infectors are called COM or EXE viruses because they infect programs with those extensions



An EXE infector can be prepending (writes itself before the original file), appending writes itself to the end of the original file), overwriting (overwrites the original file with its own code), inserting (inserts itself into gaps inside the original file), companion (renames the original file and writes itself with the original file's name) and cavity infector (writes itself between file sections of 32-bit file). An EXE infector can be memory resident and non-memory resident. Memory resident viruses stay active in memory, trap one or more system functions (usually interrupt 21h or Windows file system hooks) and infect files while they are accessed. Non-memory resident viruses search for EXE files on a hard disk and infect them. EXAMPLE: The Jerusalem virus (also called the Israeli virus) is triggered when an infected program is executed. The virus first puts the value 0E0H into register ax and invokes the DOS service interrupt (21H). If on return the high eight bits of register ax contain 03H, the virus is already resident on the system and the executing version quits, invoking the original program. Otherwise, the virus sets itself up to respond to traps to the DOS service interrupt vector. The Jerusalem virus then checks the date. If the year is 1987, it does nothing. Otherwise, if it is not a Friday and not the 13th (of any month), it sets itself up to respond to clock interrupts (but it will not infect on clock calls). It then loads and executes the file originally executed. When that file finishes, the virus puts itself in memory. It then responds to call s to the DOS service interrupt. If it is a Friday and the 13th (of any month), and the year is not 1987, the virus sets a flag in memory to be destructive. This flag means that the virus will delete files instead of infecting them.

Once in memory, the virus checks all calls to the DOS service interrupt, looking for those asking that files be executed (function 4B00H). When this happens, the virus checks the name of the file. If it is COMND.COM, the virus does nothing. If the memory flag is set to be destructive, the file is deleted. Otherwise, the virus checks the last five bytes of the file. If they are the string “MSDOS,”

the file is infected. If they are not, the virus checks the last character of the file name. If it is "M," the virus assumes that a .COM file is being executed and infects it; if it is "E," the virus assumes that a .EXE file is being executed and infects it. The file's attributes, especially the date and time of modification, are left unchanged.

Multipartite Virus

A multipartite virus is a computer virus that infects and spreads in multiple ways. Such a virus typically has two parts, one for each type. When it infects an executable, it acts as an executable infector; when it infects a boot sector, it works as a boot sector infector.

The term was coined to describe the first viruses that included DOS executable files and PC BIOS boot sector virus code, where both parts are viral themselves. Prior to the discovery of the first of these, viruses were categorized as either file infectors or boot infectors. Because of the multiple vectors for the spread of infection, these viruses could spread faster than a boot or file infector alone.

File infectors viruses are made to infect files of on the computer. File infectors spread once the user runs the infected file. The virus copies itself to locations on the computer where it can be executed; usually in RAM. The file infector will continue to infect files while granting the virus access to the infect files.

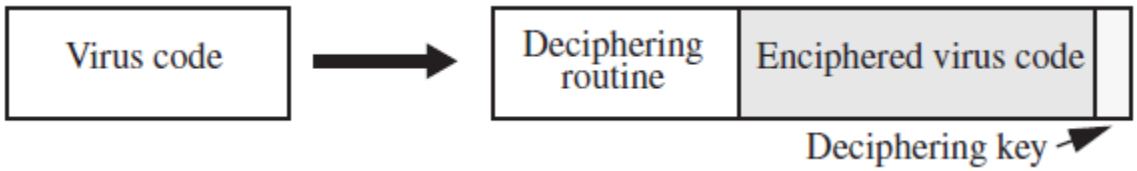
Similarly, Boot infectors spread during the boot up of a computer. Boot infectors target the critical section on the hard drive or on floppy disks in order to gain access to the computer. This enables the virus to be able to obtain complete control and/or extract any important information from your computer.

Multipartite viruses increase their chances of spreading within the computer by combining features from both the file infector and the boot infector. These viruses have the ability to infect both files and boot sectors. Because of this, the chance of the virus spreading is increased, but the virus also becomes more vulnerable to detection due to the increased number of locations an antivirus software can find the virus.

The multipartite viruses are often tricky and hard to eliminate. When all infected files have been cleaned, but the virus remains in the boot sector, files on the system will be infected again. Similarly, if the boot sectors were disinfected, but the files were still infected, then the boot sector will be re-infected. The process will continually be repeated if the virus is not removed completely from the host system.

Encrypted Virus

A virus using encryption to hide itself from virus scanners. That is, the encrypted virus jumbles up its program code to make it difficult to detect. An encrypted virus's code begins with a decryption algorithm and continues with scrambled or encrypted code for the remainder of the virus. Each time it infects, it automatically encodes itself differently, so its code is never the same. Through this method, the virus tries to avoid detection by anti-virus software. An encrypted virus should not be confused with the more recent computer viruses like crypto locker that encrypt the data on your hard drive and hold it for ransom



Polymorphic Viruses

A polymorphic virus is a virus that changes its form each time it inserts itself into another program Consider an encrypted virus. The body of the virus varies depending on the key chosen, so detecting known sequences of instructions will not detect the virus. However, the decryption algorithm can be detected. Polymorphic viruses were designed to prevent this. They change the instructions in the virus to something equivalent but different. In particular, the deciphering code is the segment of the virus that is changed. In some sense, they are successors to the encrypting viruses and are often used in conjunction with them. Consider polymorphism at the instruction level. All the instructions have exactly the same effect, but they are represented as different bit

patterns on most architectures. A polymorphic virus would insert these instructions into the deciphering segment of code.

- ✓ add 0 to operand
- ✓ or 1 with operand
- ✓ no operation
- ✓ subtract 0 from operand

Sparse Infector Virus

A sparse infector virus attempts to elude detection by performing its malicious activities only sporadically. With a sparse infector virus, the user will see symptoms for a short period, then no symptoms for a time. In some cases, the sparse infector targets a specific program but the virus only executes every 10th time or 20th time that target program executes. Or a sparse infector may have a burst of activity and then lie dormant for a period. There are a number of variations on the theme, but the basic principle is the same: to reduce the frequency of attack and thus reduce the chances for detection.

Spacefiller Virus

Many viruses take the easy way out when infecting files; they simply attach themselves to the end of the file and then change the start of the program so that it first points to the virus and then to the actual program code. Many viruses that do this also implement some stealth techniques so you don't see the increase in file length when the virus is active in memory.

A spacefiller (cavity) virus, on the other hand, attempts to be clever. Some program files, for a variety of reasons, have empty space inside of them. This empty space can be used to house virus code. A spacefiller virus attempts to install itself in this empty space while not damaging the actual program itself. An advantage of this is that the virus then does not increase the length of the program and can avoid the need for some stealth techniques. The Lehigh virus was an early example of a spacefiller virus.

Because of the difficulty of writing this type of virus and the limited number of possible hosts, cavity viruses are rare…however… A new Windows file format known as Portable Executable (PE) is designed to make loading and running programs faster. While a great goal, the implementation has the effect of leaving potentially large gaps in the program file. A cavity (spacefiller) virus can find these gaps and insert itself into them. The CIH virus family takes advantage of this new file format. There will likely be more.

Batch Files

A batch file is a kind of script file in DOS, OS/2 and Windows. It consists of a series of commands to be executed by the command line interpreter, stored in a plain text file. A batch file may contain any command the interpreter accepts interactively and use constructs that enable conditional branching and looping within the batch file, such as "if", "for", "goto" and labels. The term "batch" is from batch processing, meaning "non-interactive execution", though a batch file may not process a batch of multiple data.

Similar to Job Control Language (JCL) and other systems on mainframe and minicomputer systems, batch files were added to ease the work required for certain regular tasks by allowing the user to set up a script to automate them. When a batch file is run, the shell program (usually COMMAND.COM or cmd.exe) reads the file and executes its commands, normally line-by-line. Unix-like operating systems (such as Linux) have a similar, but more flexible, type of file called a shell script.

The filename extension .bat is used in DOS and Windows. Windows NT and OS/2 also added .cmd. Batch files for other environments may have different extensions, e.g. .btm in 4DOS, 4OS2 and 4NT related shells.

Let’s begin with a simple example, open your command prompt and change your current directory to 'desktop' by typing 'cd desktop' without quotes.

Now type these commands one by one

1. md x //makes directory 'x' on desktop

2. cd x // changes current directory to 'x'
3. md y // makes a directory 'y' in directory 'x'

We first make a folder/directory 'x', then enter in folder ‘x',then make a folder 'y' in folder 'x' . Now delete the folder 'x'. Let’s do the same thing in another way. Copy these three commands in notepad and save file as anything.bat

Now just double click on this batch file and the same work would be done , You will get a folder 'x' on your desktop and folder 'y' in it. This means the three commands executed line by line when we ran the batch file

So a batch file is simply a text containing series of commands which are executed automatically line by line when the batch file is run.

What can batch viruses do ?

They can be used to delete the windows files,format data,steal information,irritate victim, consume CPU resources to affect performance,disable firewalls,open ports,modify or destroy registry and for many more purposes.

1. Application Bomber

- @echo off // It instructs to hide the commands when batch files is executed
- :x //loop variable
- start winword
- start mspaint //open paint
- start notepad
- start write
- start cmd //open command prompt

- start explorer
- start control
- start calc // open calculator
- goto x // infinite loop

This code when executed will start open different applications like paint,notepad,command prompt repeatedly, irritating victim and ofcourse affecting performance.

2. Folder flooder

- @echo off
- :x
- md %random% // makes directory/folder.
- goto x

Here %random% is a variable that would generate a positive no. randomly. So, this code would make start creating folders whose name can be any random number.

Limitations of Batch Viruses -:

1. Victim can easily read the commands by opening batch file in notepad.
2. The command prompt screen pops up, it alerts the victim and he can stop it.

Worm

Worms are programs that replicate themselves from system to system without the use of a host file. This is in contrast to viruses, which requires the spreading of an infected host file. Although worms generally exist inside of other files, often Word or Excel documents, there is a difference between how worms and viruses use the host file. Usually the worm will release a document that already has the "worm" macro inside the document. The entire document will travel from computer to computer, so the entire document should be considered the worm. PrettyPark.Worm is a particularly prevalent example.

How does it spread?

Worms are self-replicating codes. This is the most distinct feature of a worm. Once they infect a host, they will try to find a nearby host which they can access, and copy themselves to that host. There it will perform the same actions that it performed on the original host. Worms typically cause harm to their host networks by consuming bandwidth and overloading web servers. Computer worms can also contain "payloads" that damage host computers. Payloads are pieces of code written to perform actions on affected computers beyond simply spreading the worm. Payloads are commonly designed to steal data or delete files. Some payloads even create backdoors in host computers that allow them to be controlled by other computers. Malicious parties can use networks of these infected computers ("botnets") to spread spam and perform denial-of-service attacks. Computer worms are classified as a type of computer virus, but there are several characteristics that distinguish computer worms from regular viruses. A major difference is the fact that viruses spread through human activity (running a program, opening a file, etc) while computer worms have the ability to spread automatically without human initiation. In addition to being able to spread unassisted, computer worms have the ability to self-replicate. This means that worms can create multiple copies of themselves to send to other computers. This often happens through the sending of mass emails to infected users' email contacts.

Computer Worm Examples

Computer worms have caused billions of dollars in damages over the past decade. Today, the Stuxnet, Duqu, and Flame computer worms continue to make headlines as a new breed of malware: computer worms designed for cyber warfare.

The Stuxnet virus is a computer worm discovered in June 2010. Stuxnet was created by the United States and Israel, targeting Iran's Uranium Enrichment Program. Stuxnet was created as part of a top-secret cyber war program codenamed "Olympic Games." The computer worm crashed 984 centrifuges at Iranian nuclear power plants between 2008 and 2012, setting back nuclear weapon production capabilities in Iran by about two years.

The Duqu computer worm was discovered in September 2011 and is believed to be linked to the Stuxnet virus. Duqu and Stuxnet operate very similarly and were both created by governments to target nuclear production in Iran. Rather than being used to disrupt the production of nuclear weapons, Duqu was used for stealing information. Some versions of Duqu did include a payload capable of deleting files from the host's computer.

The Flame virus was discovered in 2012 and is regarded as one of the most sophisticated computer worms ever found. Flame's code shares many similarities with the Stuxnet code, and Flame, like Stuxnet, was designed as part of a government-sponsored cyber program. While the Stuxnet computer worm was designed to sabotage nuclear weapon production, Flame is believed to have been created purely for cyber spying. Flame has infected thousands of computers since its deployment, mostly in Iran and other Middle Eastern countries.

Virus vs Worms

An important distinction between computer viruses and worms is that viruses require an active host program or an already-infected and active operating system in order for viruses to run, cause damage and infect other executable files or documents, while worms are stand-alone malicious programs that can self-replicate and propagate via computer networks, without human help.

Viruses are typically attached to an executable file or a word document. They often spread via P2P file sharing, infected websites, and email attachment downloads. Once a virus finds its way onto your system, it will remain dormant until the infected host file or program is activated, which in turn makes the virus active enabling it to run and replicate on your system.

Worms, on the other hand, don't need a host program in order for them to run, self-replicate and propagate. Once a worm has made its way onto your system, usually via a network connection or as a downloaded file, it can then make multiple copies of itself and spread via the network or internet connection infecting any inadequately-protected computers and servers on the network. Because each subsequent copy of a network worm can also self-replicate, infections can spread very rapidly via the internet and computer networks.

Morris Worm

The Morris worm or Internet worm of November 2, 1988 was one of the first computer worms distributed via the Internet. It was the first to gain significant mainstream media attention. It also resulted in the first felony conviction in the US under the 1986 Computer Fraud and Abuse Act. It was written by a graduate student at Cornell University, Robert Tappan Morris, and launched on November 2, 1988 from the computer systems of the Massachusetts Institute of Technology.

Different types of Computer Worms.

Email Worms

Spreading goes via infected email messages. Any form of attachment or link in an email may contain a link to an infected website. In the first case activation starts when the user clicks on the attachment while in the second case the activation starts when clicking the link in the email.

Known methods to spread are:

- MS Outlook services
- Direct connection to SMTP servers using their own SMTP API
- Windows MAPI functions

This type of worms is known to harvest an infected computer for email addresses from different sources.

- Windows Address Book database [WAB]
- MS Outlook address book
- Files with appropriate extensions will be scanned for email like strings

Be aware that during spreading some worms construct new sender addresses based on possible names combined with common domain names. So, the sender address in the email doesn't need to be the originator of the email.

Instant Messaging Worms

The spreading used is via instant messaging applications by sending links to infected websites to everyone on the local contact list. The only difference between these and email worms is the way chosen to send the links.

Internet Worms

Nasty ones. These ones will scan all available network resources using local operating system services and/or scan the Internet for vulnerable machines. Attempt will be made to connect to these machines and gain full access to them.

Another way is that the worms scan the Internet for machines still open for exploitation i.e. not patched. Data packets or requests will be send which install the worm or a worm downloader. If succeeded the worm will execute and there it goes again!

IRC Worms

Chat channels are the main target and the same infection/spreading method is used as above - sending infected files or links to infected websites. Infected file sending is less effective as the recipient needs to confirm receipt, save the file and open it before infection will take place.

File-sharing Networks Worms

Copies itself into a shared folder, most likely located on the local machine. The worm will place a copy of itself in a shared folder under a harmless name. Now the worm is ready for download via the P2P network and spreading of the infected file will continue.

Archivers, Encryptors, and Packers

Any number of publicly available utilities that are meant to protect data and ensure integrity can also be successfully used to protect malware during propagation, at rest, and most im portantly, from forensic analysis. Let's review, in order of evolution, archivers, encryptors, and packers, and how they are used today to infect systems.

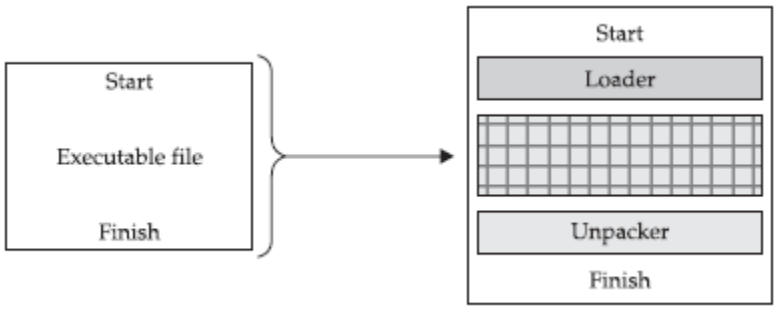
Archivers In the late 1990s, ZIP, RAR, CAB, and TAR utilities were used to obfuscate malware. In order for the archiver to run, it typically needs to be installed on the victim host unless the writer has included the utility as part of the loader. This method has become the least used of late due to the fact that in order for the malware to execute it needs to be unpacked and moved to a location on the hard disk that can be readily scanned by the antivirus engine and removed. In addition, most antivirus engines now scan deep within archived files in order to search for embedded portable executables. This method is slightly dated and not used very widely anymore due to the sophistication of antivirus scanners and their ability to scan multiple depths within archived files.

Encryptors (Malware Sample: W32.Beagle@mm! enc) These are typically employed by most software developers to protect the core code of their applications. The core code is encrypted and compressed, which makes it hard to reverse (engineer) or to identify functions within the application by hackers. Cryptovirology is a synonymous term for the study of cryptography processes used by malware to obfuscate and protect itself in order to sustain survivability. Historically, malware implemented shared-key (symmetric) encryption methods, but once the forensics community identified this method, reversing it was fairly easy, leading to the recent implementation of public key encryption.

Packers (Malware Sample:W32.Beagle@mm!enc) Almost all malware samples today implement packers in some fashion in order to bypass security programs such as antivirus or anti-spyware tools. A packer is, in simple terms, an encryption module used to obfuscate the actual main body of code that executes the true functionality of the malware. Packers are used to bypass network-detection tools during transfer and host-based protection products. There are dozens of packers publicly and privately available on the Internet today. The private and one-off packers can be the most difficult to detect since they are not publicly available and not easily identified by enterprise security products. There is a distinct difference between packers and archiving utilities. The common computer user does not generally employ packers. Packers generally protect executables and DLLs without the need for any preinstalled utility on the victim host. Just like hackers skill

levels, packers also have varying levels of sophistication that come with numerous functionality options. Packers often protect against antivirus protections and also increase the ability of the malware to hide itself. Packers can provide a robust set of features for hackers such as the ability to detect virtual machines and crash when within them; generate numerous exceptions, leveraging polymorphic code to evade execution prevention; and insert junk instructions that increase the size of the packed file, thereby making detection more difficult. You will typically find ADD, SUB, XOR, or calls to null functions within junk instructions to throw off forensic analysis. Typically, you will also find multiple files packed or protected together such as the executable files; other executable files will be loaded in the address space of the primary unpacked file.

The following is a simple example of a packer's process:



The most powerful part of using a packer is the malware never needs to hit the hard disk. Everything is run as in-process memory, which can generally bypass most antivirus and host-based security tools. With this approach, if the packer is known, an antivirus engine can identify it as it unpacks the malware. If it is a private or new packer, the antivirus engine has no hope of preventing the malware from executing; the game is lost and no security alerts are triggered for an administrator to act upon.

Trojan Horse

Trojan horse is a term for a program that looks benign but actually has a malicious purpose. We have already seen viruses that are delivered via a Trojan horse. You might receive or download a

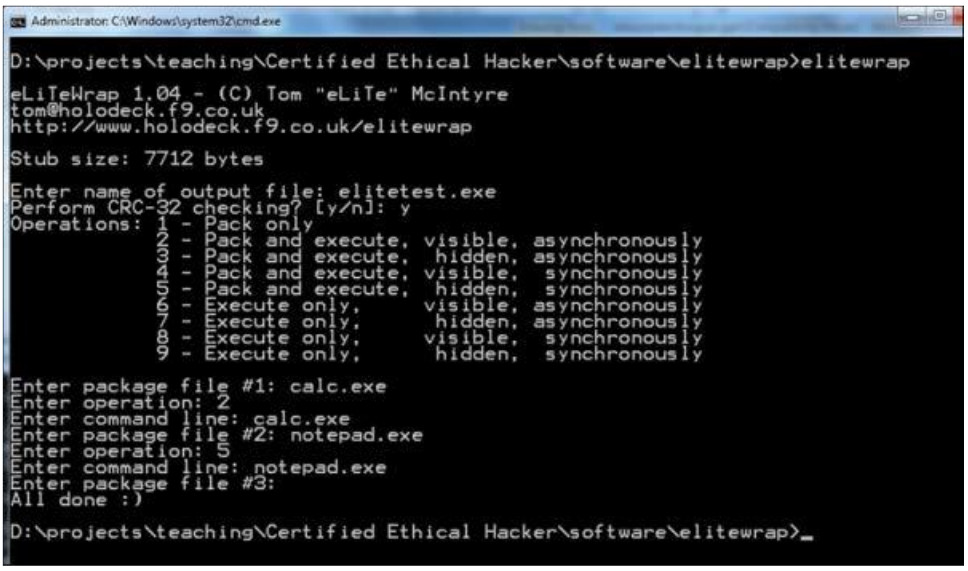
program that appears to be a harmless business utility or game. More likely, the Trojan horse is just a script attached to a benign-looking email. When you run the program, or open the attachment, it does something else other than or in addition to what you thought it would. It might

- Download harmful software from a website.
- Install a key logger or other spyware on your machine.
- Delete files.
- Open a backdoor for a hacker to use.

It is common to find combination virus plus Trojan horse attacks. In those scenarios, the Trojan horse spreads like a virus. The MyDoom virus opened a port on your machine that a later virus, doomjuice, would exploit, thus making MyDoom a combination virus and Trojan horse. A Trojan horse could also be crafted especially for an individual. If a hacker wished to spy on a certain individual, such as the company accountant, he could craft a program specifically to attract that person's attention. For example, if he knew the accountant was an avid golfer, he could write a program that computed handicap and listed best golf courses. He would post that program on a free web server. He would then email a number of people, including the accountant, telling them about the free software. The software, once installed, could check the name of the currently logged-on person. If the logged on name matched the accountant's name, the software could then go out, unknown to the user, and download a key logger or other monitoring application. If the software did not damage files or replicate itself, then it would probably go undetected for quite a long time. There have been a number of Trojan horses through the years. One of the earliest and most widely known was Back Orifice . Such a program could be within the skill set of virtually any moderately competent programmer. This is one reason that many organizations have rules against downloading any software onto company machines. However, it is important to remember that those creating virus attacks tend to be innovative people. It is also important to note that creating a Trojan horse does not require programming skill. There are free tools on the Internet, such as EliteWrapper, that allow someone to combine two programs, one hidden and one not. So one could easily take a virus and combine it with, for example, a poker game. The end user would only

see the poker game, but when it was run it would launch the virus. Another scenario to consider is one that would be quite devastating. Without divulging programming details, the basic premise will be outlined here to illustrate the grave dangers of Trojan horses.

Imagine a small application that displays a series of unflattering pictures of Osama Bin Laden. This application would probably be popular with many people in the United States, particularly people in the military, intelligence community, or defense-related industries. Now assume that this application simply sits dormant on the machine for a period of time. It need not replicate like a virus because the computer user will probably send it to many of his associates. On a certain date and time, the software connects to any drive it can, including network drives, and begins deleting all files. If such a Trojan horse were released "in the wild," within 30 days it would probably be shipped to thousands, perhaps millions, of people. Imagine the devastation when thousands of computers begin deleting files and folders. This scenario is mentioned precisely to frighten you a little. Computer users, including professionals who should know better, routinely download all sorts of things from the Internet, such as amusing flash videos and cute games. Every time an employee downloads something of this nature, there is the chance of downloading a Trojan horse. One need not be a statistician to realize that if employees continue that practice long enough, they will eventually download a Trojan horse onto a company machine. If they do, hopefully the virus will not be as vicious as the theoretical one just outlined here. Because users usually install Trojan horses themselves, the security countermeasure for this attack is to prevent downloads and installations by end users. From a law enforcement perspective, the investigation of a crime involving a Trojan horse would involve a forensic scan of the computer hard drive, looking for the Trojan horse itself. There are a number of tools, some free for download, that will help a person create a Trojan horse. For example, eLiTeWrap is easy to use. Essentially, it can bind any two programs together. Using a tool such as this one, anyone can bind a virus or spyware to an innocuous program such as a shareware poker game. This would lead to a large number of people downloading what they believe is a free game and unknowingly installing malware on their own system.



- Trojan-Banker
  Trojan-Banker programs are designed to steal your account data for online banking systems, e-payment systems, and credit or debit cards.

- Trojan-DDoS
  These programs conduct DoS (Denial of Service) attacks against a targeted web address. By sending multiple requests – from your computer and several other infected computers – the attack can overwhelm the target address… leading to a denial of service.

- Trojan-Downloader
  Trojan-Downloaders can download and install new versions of malicious programs onto your computer – including Trojans and adware.

- Trojan-Dropper
  These programs are used by hackers in order to install Trojans and / or viruses – or to prevent the detection of malicious programs. Not all antivirus programs are capable of scanning all of the components inside this type of Trojan.

- Trojan-FakeAV
  Trojan-FakeAV programs simulate the activity of antivirus software. They are designed to extort money from you – in return for the detection and removal of threats… even though the threats that they report are actually non-existent.

- Trojan-GameThief
  This type of program steals user account information from online gamers.

- Trojan-IM
  Trojan-IM programs steal your logins and passwords for instant messaging programs – such as ICQ, MSN Messenger, AOL Instant Messenger, Yahoo Pager, Skype, and many more.

- Trojan-Ransom
  This type of Trojan can modify data on your computer – so that your computer doesn't run correctly or you can no longer use specific data. The criminal will only restore your computer's performance or unblock your data, after you have paid them the ransom money that they demand.

- Trojan-SMS
  These programs can cost you money – by sending text messages from your mobile device to premium rate phone numbers.

- Trojan-Spy
  Trojan-Spy programs can spy on how you're using your computer – for example, by tracking the data you enter via your keyboard, taking screen shots, or getting a list of running applications.

- Trojan-Mailfinder
  These programs can harvest email addresses from your computer.

- Other types of Trojans include:
  - Trojan-ArcBomb
  - Trojan-Clicker
  - Trojan-Notifier
  - Trojan-Proxy
  - Trojan-PSW

A remote administration tool (RAT) is a piece of software that allows a remote "operator" to control a system as if they have physical access to that system. While desktop sharing and remote administration have many legal uses, "RAT" software is usually associated with criminal or malicious activity. Malicious RAT software is typically installed without the victim's knowledge, often as payload of a Trojan horse, and will try to hide its operation from the victim and from security software.

The operator controls the RAT through a network connection. Such tools provide an operator the following capabilities:

- Screen/camera capture or image control
- File management (download/upload/execute/etc.)
- Shell control (from command prompt)
- Computer control (power off/on/log off if remote feature is supported)
- Registry management (query/add/delete/modify)
- Hardware Destroyer (overclocker)
- Other software product-specific functions

Its primary function is for one computer operator to gain access to remote PCs. One computer will run the "client" software application, while the other computer(s) operate as the "host(s)".

The basic methodology of using a trojan is as follows: -

1. Attacker creates an executable file of size in kbs. This is server part of Trojan and mostly called as server.exe
2. Attacker might hide this server.exe behind any genuine file like a song or image. Attacker gives this file to victim and victim is supposed to double click on it.

3. As victim run that server part, a port on victim's computer gets opened and attacker can control his PC sitting remotely in any part of the world through the control panel (client part). Attacker can do anything with victim's computer remotely that victim himself can do on his computer.

Two different methods of working of Trojan.

1. Direct Connection : In this method, after the server part has been installed on victim's machine, the attacker enters the public IP address assigned to victim's computer for making a connection to it. But limitations of direct connection is that public IP address is most probably dynamic and

gets changed everytime one disconnects and reconnects. So attacker needs to find out IP address of victim each time.Moreover the incoming connection like this is usually restricted by firewall.

The main limitation of direct connection is that you can not access the victim who is behind a router or a network beacuse victim's machine is not assigned public/external/wan IP. It is only assigned private/internal/lan IP which is useless or meaningless for computers outside that network. The wan IP belongs to his router.

Reverse Connection: In this method, attacker enters his own IP address in server part while configuring it. So, when the server part is installed on victim's computer, it automatically makes connection with client part that is attacker. Also, the firewall in victim's machine would not restrict to outgoing connections. Problem in this case is same that attacker's IP is also dynamic. But this can be over come easily. Attacker actually enters a domain name in server part which always points to his dynamic IP.

What Is a Rootkit?

A rootkit is a clandestine computer program designed to provide continued privileged access to a computer while actively hiding its presence. The term rootkit is a connection of the two words "root" and "kit." Originally, a rootkit was a collection of tools that enabled administrator-level access to a computer or network. Root refers to the Admin account on Unix and Linux systems, and kit refers to the software components that implement the tool. Today rootkits are generally associated with malware – such as Trojans, worms, viruses – that conceal their existence and actions from users and other system processes.

What Can a Rootkit Do?

A rootkit allows someone to maintain command and control over a computer without the computer user/owner knowing about it. Once a rootkit has been installed, the controller of the rootkit has the ability to remotely execute files and change system configurations on the host

machine. A rootkit on an infected computer can also access log files and spy on the legitimate computer owner's usage.

Case Study

In January 2008, a new type of rootkit appeared in the wild, stealing financial data by installing keyloggers on computers and monitoring when users entered their username and password for many European banks. This new rootkit was the most malicious of its kind ever seen. Invisible to all anti-rootkit and anti-malware utilities, including those from McAfee, Symantec, and even Kaspersky, this rootkit downloads malware that logs all keystrokes typed into the computer. Between December 12, 2007, and January 7, 2008, iDefense, a security firm owned by Verisign, detected approximately 5000 machines infected with the rootkit in Europe. The rootkit, once installed, embeds itself into the Master Boot Record (MBR) of the computer.

The MBR is the first 512 bytes of the computer's primary hard drive. The BIOS of the computer tells the CPU to execute the machine code written to the first 512 bytes. This machine code, commonly referred to as a boot manager, typically starts the operating system loaded on the computer and directs it to access the first partition available on the system.

The boot manager can be replaced by other code if the operating system allows the first 512 bytes of the hard drive to be overwritten. Microsoft Windows allows the MBR to be overwritten by applications that are executed by an administrative user. Users infect themselves when they access websites intended to spread the virus such as various pornographic and illegal software (warez) sites. The rootkit, named Mebroot, exploits users running a copy of Internet Explorer that is vulnerable to an exploit. Once exploited, the rootkit downloads a 450kb (rather large) file that, when executed, stores itself in the last few sectors of the hard drive and writes a copy of its rootkit boot manager to the MBR and executes itself. Since the rootkit was written to the MBR, when the system reboots, the rootkit will be executed before the operating system, thereby ensuring it is loaded first and can re-infect the computer.

What makes this rootkit even worse is that researchers at F-Secure and Symantec have proof that the rootkit was "beta" tested in early November 2007 to ensure it functioned properly. The date- and timestamps on the executables found in the wild indicate that in November 2007, a specific domain on the Internet started to spread an early version of what would become the Mebroot rootkit. After the beta release, two additional waves of Mebroot with some amazing capabilities were released to the world. Besides being one of the first rootkits to infect the MBR, Mebroot is the pinnacle of professional rootkit development. The methods used to execute processes, hide network traffic, and prevent detection are advanced and still, as of this writing, very effective at evading detection. Mebroot innovates in three main areas: stealth disk access, firewall bypassing, and backdoor communication. Each of these capabilities is implemented within the Microsoft Windows kernel, a portion of the operating system usually reserved for drivers that manage your network card or graphics card. The amount of skill needed to implement these capabilities at the kernel level itself is exemplary but even more so for a rootkit where traditional rootkit developers do nothing but copy code from other authors and websites and change a few things.

Disk Access

Traditional rootkits prevent access to portions of the hard drive by intercepting the functions executed by applications such as `CreateFile()`. Mebroot is different. Instead of just intercepting the function calls by overwriting certain portions of the DISK.SYS driver in memory, which would be detectable, the Mebroot rootkit overwrites all functions within the DISK.SYS driver and installs a wrapper driver that calls the DISK. SYS functions to ensure that behavioral products such as host-based intrusion prevention systems do not prevent it from infecting DISK.SYS. As an extra measure, the rootkit also starts a "watchdog" thread that checks every couple of seconds to ensure that the rootkit's stealth capabilities are still installed on the system. If they are removed, the rootkit will reinstall them.

Firewall Bypassing

Rootkits need a way to covertly allow themselves and any malware they work with to access the network to request web pages and communicate with its Command and Control (C&C) server. Of course, if the rootkit does not hide its communication, antirootkit tools such as firewalls and host intrusion prevent systems (HIPS) may detect it. Until Mebroot, most rootkits simply worked by creating and installing a driver similar to a network card driver within the Windows' kernel's network interface, NDIS. Mebroot didn't want to be detected so the developers did not use this method. Instead, the developer wrote a set of algorithms to find hidden and undocumented functions within Microsoft's NDIS, which allows the rootkit to communicate with NDIS without installing a driver. This method, although stealthy, requires that the rootkit implement its own TCP/IP stack to communicate with other devices on the Internet. Writing your own TCP/IP stack is difficult, which goes to show how focused the authors were during development and the lengths that rootkit developers must go to to remain undetected.

Backdoor Communication

Mebroot utilizes advanced firewall bypassing techniques to covertly communicate with Command and Control (C&C) servers on the Internet and process commands from the owner of the botnet. Researchers, however, have only seen one of the multiple commands the rootkit can interpret in the wild (the INST command, which installs various malware). First, the rootkit connects to a random C&C server, building a domain name using the current time and date as well as a variety of hard-coded domains. Once the rootkit resolves a DNS name to an IP address, the rootkit sends an encrypted packet to the IP address to "ping" the C&C server to make sure it responds to its encrypted communication. The rootkit uses an encryption algorithm based on SHA-1, an industry standard but one that uses a very weak and easily decipherable key, which has allowed researchers to decrypt the packets. To add additional complexity, however, the decrypted packet actually contains data that is then encrypted using a different encryption scheme found in other malware.

Once the C&C server responds to the rootkit, the C&C server can tell the rootkit to execute one of four commands:

- Install a DLL into any process or install a new version of Mebroot.
- Uninstall a user-mode DLL or uninstall Mebroot.
- Instruct a trusted process to launch new processes by fi lename.
- Execute any driver in kernel mode.

The ability to uninstall the rootkit is further evidence that the rootkit was developed and tested by professionals, as an uninstall function can aid in debugging and creating a rootkit. Once the command is received, the rootkit will execute each command on the system using very detailed instructions (too detailed for this case study!) to ensure that antirootkit technologies do not prevent the command from executing. For example, the rootkit uses a built-in system call system (similar to what an operating system does!) to rewrite custom DLLs that it is told to execute on the system.

Intent

What could be so beneficial that the Mebroot developers would spend potentially months of time developing such an advanced rootkit? Money. The Mebroot rootkit installs and executes malware delivered by the C&C server to infect hosts. This malware can record keystrokes, sniff HTTP and HTTPS requests, and inject arbitrary HTML into websites, particularly banking sites. These features enable many different types of fraud, including identity theft, click-fraud, and the theft of bank accounts. As of this writing, a new wave of Mebroot has been found in the wild with even more innovative stealth techniques. The Mebroot rootkit is one of the most advanced rootkits the public has ever seen. Written by professionals, effective, and hard to remove, this rootkit delivers malware that is used to steal financial information such as bank account and credit card numbers. Mebroot, with all of its advanced capabilities, is just the start of the new evolution of rootkits that will propel what was previously easily cleaned or ineffective malware into a new echelon of capabilities. Now, not all is lost. Mebroot can be removed, and the easiest way to remove this rootkit is to run the fixmbr command from within the Windows recovery console, which is available by booting of the Windows XP CD (included with all Windows installations). This overwrites the

rootkit's entry on MBR with a standard Windows MBR. Also some of the latest BIOS settings allow you to make the MBR read-only. If set to read-only, any modification to the MBR will cause a BIOS warning.

Rootkits and their functionality have changed over the years. The functionality of these applications has led to a very fast adoption rate by the underground community, so it helps to understand where rootkits have come from, why they have adapted to their environment, and what attackers will be doing with them in the future. The predecessor of the first rootkit was actually not a rootkit at all but a set of applications that removed evidence of an intrusion from a machine. So-called log cleaner kits were found as early as 1989 on hacked systems and helped attackers cover their tracks as they broke into system after system. These automated applications would be executed by an attacker as soon as he or she got administrative access to a server. The tools would seek out the various log files that stored which user was logged in and what commands that user executed. Once these files were found, the applications would open the files and either strategically remove certain logs or delete the entire file. While log cleaners helped cover up initial access to a system, attackers wanted to always be protected from a system administrator finding out that they had been on the company's server. This requirement led to the creation of the first-generation rootkit. The first-generation served one major purpose—execute commands for an attacker without being seen. Traditionally, an attack would consist of the attacker exploiting a vulnerable network service such as inetd, a Unix application that connects network sockets to applications, cleaning the logs, and then adding a new user to the system so the attacker could access the system again. This backdoor account is common even today as attackers want to maintain access to a system. The problem with adding a new user is that administrators can see it. To prevent this, the first-generation involved the teaming of log cleaners and new versions of common command-line tools in Unix such as `ls`, which lists files in a directory, and `ps`, which lists what programs are running on the system. These new versions removed the newly created backdoor user's files and processes from the tools' output.

MAINTAIN ACCESS

Maintaining access to a hacked system is very important for an attacker. With the ability to log back into a server with full administrative privileges, the attacker can leverage the server for other attacks, store data, or host a malicious website. Rootkits maintain access by installing either local or remote backdoors. A local backdoor is an application that, once executed, will give normal users full administrative privileges on the system. Local backdoors were common in early rootkit development as many attackers of systems were actually normal users trying to elevate their privileges. Furthermore, attackers would keep a local backdoor around, in addition to a local backdoor user account, just in case the remote backdoors didn't work. Remote backdoors were generally the best way to go. Early rootkits had a variety of remote backdoors. The stealth and sophistication of the backdoors within the rootkits is what set each rootkit apart. The types of remote backdoors generally fall into three categories: Network Socket Listener, Trojan, or covert channels.

STEALTH: CONCEAL EXISTENCE

The second major feature of rootkits is their ability to conceal any evidence of their existence on the system. As we mentioned, rootkits evolved from programs that attackers used to remove the logs on a system they broke into. As rootkits started to morph into those that provided continual "root" access to the system, a new requirement to hide any files or registry keys that the rootkit needed to operate became essential. If the rootkit hid these items, the system administrator and anti-rootkit tools would have a much harder time detecting the rootkit. Most rootkits will hide files they generate, any files specified by the user of the rootkit, and any network connections the rootkit generates. What to hide is usually specified in a configuration file or hardcoded into the rootkit itself. The latest generations of rootkits use their stealth abilities to help other malware such as programs that steal usernames, passwords, and bank account information by hiding them from users and anti-malware tools. The teaming of malware with rootkits has caused rootkit developers to improve the quality and effectiveness of their stealth techniques dramatically. When rootkits were first detected in Unix environments, they usually only implemented their hiding capabilities using one method; for example, they would filter out files when the tool `ls` was used

but not when a custom tool that read files from the file system was executed. The latest Windows rootkits, such as the one used by Rustock.C, use multiple methods to ensure nothing is missed. Stealth is a major component of any rootkit. Why is stealth so important for us to talk about? Simply because most rootkit detection tools detect the changes the stealth functionality make to the system to find the rootkit itself!

TYPES OF ROOTKITS

There are generally two types of rootkits: user-mode and kernel-mode. User-mode rootkits run within the environment and security context of a user on the system. For example, if you were logged into your workstation as the user *bwilson* and did not have administrative privileges, the rootkit will filter and give backdoor access to all applications running under the *bwilson* account. Generally, most user accounts also have administrative privileges so a user-mode rootkit can also prevent system-level processes such as Windows services from being affected by its stealth functionality. Although this book primarily focuses on Windows malware and rootkits, there is another type of rootkit in the Unix world that is very similar to a user-mode rootkit. This rootkit, commonly referred to as a library rootkit, filters calls that applications make to various shared system libraries. Because they are not tied directly to a specific username, these rootkits can be more effective than standard user-mode rootkits but not as effective or hard to remove as kernel-mode rootkits.

Kernel-mode rootkits operate within the operating system at the same level as drivers for hardware such as your graphics card, network card, or mouse. Writing a rootkit for use within the kernel of an operating system is much more difficult than writing a user-mode rootkit and requires a much higher skill set from the attacker to implement. Furthermore, since many operating systems change portions of their kernel with updates and new versions, kernel rootkits don’t work for all versions of Windows. Since the rootkit operates like a driver does in the kernel, it also has the ability to increase the instability of the operating system. Normally, this is how most

performance, the appearance of blue screens, or other errors that cause the system to reboot spontaneously.

TIMELINE

Rootkits have evolved over time. Starting off as a simple set of tools to help maintain access to a machine, they have evolved into vicious applications that hide themselves, other files, are difficult to remove, and aid other malware. The following is a quick timeline to give you an understanding of the rootkit's evolution:

• Late 1980s First log cleaners found.
• 1994 First SunOS rootkits found.
• 1996 First Linux rootkits appear in the wild.
• 1997 Loadable kernel module–based rootkits are mentioned in *Phrack.*
• 1998 Silvio Cesare releases fi rst non-LKM kernel-patching rootkit code. Back Orifi ce, a fully featured backdoor for Windows, is released.
• 1999 NT Rootkit, the fi rst Windows rootkit, is released by Greg Hoglund.
• 2000 t0rnkit libproc rootkit/Trojan is released.
• 2002 Sniffer backdoors start to show up in rootkits. Hacker Defender is released, becoming one of the most used Windows rootkits.
• 2004 Majority of rootkit development in Unix stops as the focus shifts to Windows. FU rootkit is released and introduces a new technique to conceal
processes.
• 2005 Sony BMG rootkit scandal occurs. First use of rootkit technology for commercial use.
• 2006 Rootkits become part of almost every major worm and virus. Virtual rootkits start to be developed.
• 2008 After two years of relatively no new technology, rootkits in the wild start to leverage the boot process to install themselves by adapting code from eEye
Bootroot rootkit.

Bots, Botnets and Zombies

News about internet crimes often mentions 'bots', 'zombies', and 'botnets'. It's not hard to figure out from the context that these are computer or network security threats. But what exactly are they, how do they work, and what damage do they cause?

A 'bot', short for robot, is a type of software application or script that performs tasks on command like indexing a search engine, and they are really good at performing repetitive tasks. Bad bots perform malicious tasks allowing an attacker to take complete control over an affected computer for the criminal to control remotely. Once infected, these machines may also be referred to as 'zombies'. Taking over one computer is useful, but the real value to a criminal comes from collecting huge numbers of computers and networking these (a botnet) so they can all be controlled at once and perform large scale malicious acts. As of August 2011 there are between 100-150 million computers worldwide (out of 600 million PCs on the Internet) infected with bots and under the control of hackers. These computer owners unwittingly put everyone at risk, and most would be shocked to learn that the spam you're receiving is coming from thousands or even millions of computers just like (and including) theirs. The real owners of those computers can still use them, and they are probably unaware of anything being wrong except perhaps they think their computer seems slow at times.

One botnet, called Rustock, was disabled through collaboration between industry and law enforcement in March of 2011. This botnet had approximately 1 million infected computers networked together, and was capable of sending up to 30 billion spam emails a day. This botnet was so large that when it was taken down, global spam volumes instantly dropped by 30 percent.It only takes minutes for an unprotected, internet connected computer to be infected with malicious software and turned into a bot, underscoring the critical need for every computer and smartphone user to have up-to-date security software on all their devices.

Cybercriminals make money from their botnets in several ways:

They may use the botnets themselves to send spam, phishing, or other scams to trick consumers into giving up their hard-earned money. They may also collect information from the bot-infected machines and use it to steal identities, run up loan and purchase charges under the user's name. They may use their botnets to create denial-of-service (DoS) attacks that flood a legitimate service or network with a crushing volume of traffic. The volume may severely slow down the company's service or network's ability to respond or it may entirely overwhelm the company's service or

network and shut them down. Revenue from DoS attacks comes through extortion (pay or have your site taken down) or through payments by groups interested in inflicting damage to a company or network. These groups include "hacktivists" — hackers with political agendas as well as foreign military and intelligence organizations.

Cybercriminals may also lease their botnets to other criminals who want to send spam, scams, phishing, steal identities, and attack legitimate websites, and networks. Don't let your computer become a bot If you have not installed security software and ensured that it is turned on, and kept up-to-date your machine is likely infected with all kinds of malicious software, including bots. The best protection is to set your anti-virus and anti-spyware programs to automatically update, and to install every patch that your operating system and browser make available.

Even the most up-to-date protection tools cannot protect you from everything; there is still some risk because the developers of malware are always looking for new ways to get around security measures, and there is the risk of infection because of actions you, or another person who used the computer, take. A common user risk comes through downloading content from unknown sites OR from friends that don't have up-to-date protections. The intent may not be malicious at all, but if content comes from an unprotected computer it may well be infected. By downloading the content, you bring the malicious code past your security checkpoints where they can try to clean the malware of your machine, but they have no way of defending against it being downloaded in the first place. Always use extreme caution when downloading information or files from someone whose computer is not protected.

How do botmasters connect to your computer to control the malware on it?

In short, outsiders can't easily connect into your network by default, even if you want them to. So, how do botmasters connect to your computer to control the malware on it? The answer is staggeringly simple: the crooks don't call you and tell you what to do.You call them and ask for instructions. Just like Windows Update, which connects to Microsoft's servers to check for patches. And just like your webmail, which gets pulled down by your browser when you're logged

in, rather than pushed to your computer by a mail-sending server. A good firewall and anti-virus combination can still protect you, of course, by keeping track of what connections your computer makes, and which programs make them, and what gets downloaded.

After all, if your computer is going to be sending 100,000 spams over the next 24 hours, those few minutes waiting to get started will make no difference to the outcome. On the other hand, the crook doesn't have to keep trying to contact your computer if he doesn't get through the first time, for example because you're asleep and so is your laptop. The next time you turn it on, it will get busy with all its outstanding tasks automatically, including catching up on its backlog of spam sending.

The process by which bots fetch their what-to-do-next instructions is known as *command-and-control*(abbreviated C&C, or sometimes C2), and the places bots connect to are known, unsurprisingly, as *C&C servers*. Bots that use the same C&C network, and can therefore be controlled simultaneously by a single botmaster, make up a *botnet*, short for "robot network."

Logic bombs

Logic bombs are types of malware that are waiting for something to happen. They're waiting for this pre-defined event to occur. At that point, something goes into effect. These can be really, really difficult to find. Obviously, they're not a virus. It's not something that's known by anti-malware or anti-spyware, and if it goes off, the people that are writing these logic bombs are generally destroying things.

KEVIN POULSEN SECURITY 01.29.09 10:41 AM

FANNIE MAE LOGIC BOMB WOULD HAVE CAUSED WEEKLONG SHUTDOWN

A logic bomb allegedly planted by a former engineer at mortgage finance company Fannie Mae last fall would have decimated all 4,000 servers at the company, causing millions of dollars in damage and shutting down Fannie Mae for a least a week, prosecutors say.

Unix engineer Rajendrasinh Babubha Makwana, 35, was indicted (.pdf) Tuesday in federal court in Maryland on a

This one was at Fannie Mae, so a very, very large organization. He set, this is someone who had been dismissed by his job that set a logic bomb to completely disrupt over 4,000 servers at their organization. Now in this particular case, fortunately the logic bomb was found before it went off and so the entire script that was built to really create problems never really created a problem for the organization.

// ENTERPRISE APPLICATIONS

Ex-UBS Systems Admin Sentenced To 97 Months In Jail

Roger Duronio was found guilty of computer sabotage and securities fraud for writing, planting, and disseminating malicious code that took down up to 2,000 servers.

The former systems administrator convicted this past summer of launching an attack on UBS PaineWebber four years ago was sentenced to 97 months in jail in U.S. District Court in Newark, N.J., on Wednesday.

Roger Duronio, 63, of Bogota, N.J., stood quietly and didn't react as Judge Joseph Greenaway Jr. handed down the sentence. "This is a sophisticated crime," said the judge. "This wasn't an instance when an individual argues that 'I had a bad day and I made a mistake.' Its undoubtedly that Mr. Duronio, having felt wronged, came up with an elaborate, sophisticated scheme to take down a company." Judge Greeaway added that he was struck by Duronio's attempt to not only disrupt the company but to derive financial benefit from it.

In UBS, the system administrator was fired and then he put a logic bomb onto the systems. And one of the things that he did that made this one especially bad is after he put the bomb that was going to take out a huge part of this organization– this is a bank, a financial organization– he went to a stockbroker and got put options, which means if the stock went down, he would make money. So obviously, this is a very, very big problem.

Adware

When your computer gets infected with adware, you almost recognize it immediately. Suddenly you've got tons of popups on your screen. Your eyeballs are now seeing tons of ads being thrown at it, and that's because there's usually something that's hooked into your browser or another piece of malware running on your computer, that's simply popping up ads and feeding those ads to you. Your computers turned now into one big advertisement for many, many different things.

This can also, of course, cause performance issues for you. Having this information come across the network, this malware's probably communicating back to the mother-ship the things that you may have clicked on, the things you may have seen. This may be something that was installed accidentally. It could be something that you clicked on, and not realizing it, that that was malware. It may be presented to you as somewhat of a Trojan horse, or it may be something that's installed

along with other pieces of software. It may be that the software manufacturer had no idea that these bad guys had stuck some adware along with it.

When the term is used in this way, the severity of its implication varies. While some sources rate adware only as an "irritant", others classify it as an "online threat" or even rate it as seriously as computer viruses and trojans. The precise definition of the term in this context also varies. Adware that observes the computer user's activities without their consent and reports it to the software's author is called spyware. However most adware operates legally and some adware manufacturers have even sued antivirus companies for blocking adware.

A new wrinkle is adware (using stolen certificates) that disables anti-malware and virus protection; technical remedies are available. Adware has also been discovered in certain low-cost Android devices, particularly those made by small Chinese firms running on Allwinner systems-on-chip. There are even cases where adware code is embedded deep into files stored on the system and boot partitions, to which removal involves extensive (and complex) modifications to the firmware.

Superfish: Superfish was an advertising company that developed various advertising-supported software products based on a visual search engine. Superfish's software has been described as malware or adware by many sources. The software was bundled with various applications as early as 2010, and Lenovo began to bundle the software with some of its computers in September 2014. On February 20, 2015, the United States Department of Homeland Security advised uninstalling it and its associated root certificate, because they make computers vulnerable to serious cyberattacks, including interception of passwords and sensitive data being transmitted through browsers.

Spyware

Spyware is software that is specifically designed to watch what you're doing. It's spying on your browsing. It's spying on what you're typing in at the keyboard. It's trying to identify a lot of different things about you, and that's because the software these days can present advertising

that's tailored to you. It can provide private information back to someone else that can then use that for identity fraud.

These days there's really, really big money in getting your identity, and getting your private information, so that people can open up other lines of credit, credit cards, open up bank accounts with your personal information, or even worse, go into your existing credit cards, and your existing bank accounts, to gather the money directly from you.

Online Attackers Online attackers' primary interest in spyware is using it to steal personal information for financial crimes such as carding (illicit trafficking in stolen credit card and credit card information) and identity theft, or to sell that information to someone else who then executes more traditional financial crimes.

Marketing Organizations Marketing organizations are interested in personal information such as email addresses, online shopping and browsing habits, keywords in search queries, and other personal and trend-related information that can be used to execute marketing campaigns like spam, spim (unsolicited messages received via instant messaging systems), browser popups, home page hijacking (changing the default web address for a user's browser), and more. Spying by a Trusted Insider Trusted insiders include those who have physical access to computer systems for legitimate purposes. Some examples are employees, contractors, temporary workers, and cleaning crews.

How Spyware Operates

Spyware tracks online activity looking for web sites visited, financial data or identity data such as credit card numbers on screen or entered into form fields, browsing and online purchasing habits, and authentication credentials. When keywords of interest like names of banks, online payment systems, or pornographic web sites are observed, the spyware starts its data collection process.

Email Addresses

Email addresses can be harvested from an infected user's computer and marketed for use in spam mailing lists. Common techniques for harvesting email addresses and other contact information includes enumerating email applications' address books, monitoring incoming and outgoing network packets related to email, and scanning files on the system's disks for strings that match the format of an email address.

Windows Protected Store

Windows contains a service called the Protected Store. Its purpose is to provide encrypted storage for sensitive data. The following are some examples of data that might be in the PStore:

- Outlook passwords
- Passwords for web sites
- MSN Explorer passwords
- IE AutoComplete passwords
- IE AutoComplete fields
- Digital certificates

Even though the PStore is encrypted, access to it is indirectly controlled by the data owner's login credentials. Since most spyware runs under the security profile of the user who is logged on, spyware can harvest this information

Clipboard Content

The system clipboard often contains sensitive information. Some common examples include product registration codes and user credentials that are copied and pasted into login forms. Other information that might be found in the system clipboard buffer includes sections of potentially sensitive data from recently modified documents or personal information about you or your associates that could be used in crimes related to identity theft.

The Keys You Press

Key logging is one of the first spyware techniques used to capture sensitive data from a system. Both hardware and software key loggers exist. Hardware devices usually slip inline between the keyboard cable and computer. Modern key logging hardware is small and unobtrusive and has even been hidden inside the physical keyboard casing, making it almost impossible to detect.

Impact of Spyware

Spyware can cause people to lose trust in the reliability of online business transactions. Similar to the problem of counterfeit currency in the physical world, spyware undermines confidence in online economic activity. Consumers' willingness to participate in online monetary transactions decreases for fear of personal financial loss. Vendors lose confidence that the person making the purchase is who they say they are and not actually a criminal using a stolen identity or illicit funds. In efforts to manage the risk, vendors and financial institutions often implement additional verification and other loss prevention programs at increased operational cost. Even when financial organizations cover an individual's loss from online fraud, these costs plus the overhead required to administer loss prevention programs are eventually passed back to consumers in the form of higher service fees, interest rates, or other price increases on the goods and services consumed. As a result, growth rates in commerce are slowed, costs increase, and demand shrinks.

Rogue Security Software

Rogue security software is a form of malicious software and Internet fraud that misleads users into believing there is a virus on their computer, and manipulates them into paying money for a fake malware removal tool (that actually introduces malware to the computer). It is a form of scareware that manipulates users through fear, and a form of ransomware. Rogue security software has become a serious security threat in desktop computing since 2008.

Rogue security software mainly relies on social engineering (fraud) to defeat the security built into modern operating system and browser software and install itself onto victims' computers. A website may, for example, display a fictitious warning dialog stating that someone's machine is

infected with a computer virus, and encourage them through manipulation to install or purchase scareware in the belief that they are purchasing genuine antivirus software. Some rogue security software, however, propagate onto users' computers as drive-by downloads which exploit security vulnerabilities in web browsers, PDF viewers, or email clients to install themselves without any manual interaction.

More recently, malware distributors have been utilizing SEO poisoning techniques by pushing infected URLs to the top of search engine results about recent news events. People looking for articles on such events on a search engine may encounter results that, upon being clicked, are instead redirected through a series of sites before arriving at a landing page that says that their machine is infected and pushes a download to a "trial" of the rogue program. A 2010 study by Google found 11,000 domains hosting fake anti-virus software, accounting for 50% of all malware delivered via internet advertising

Keyloggers

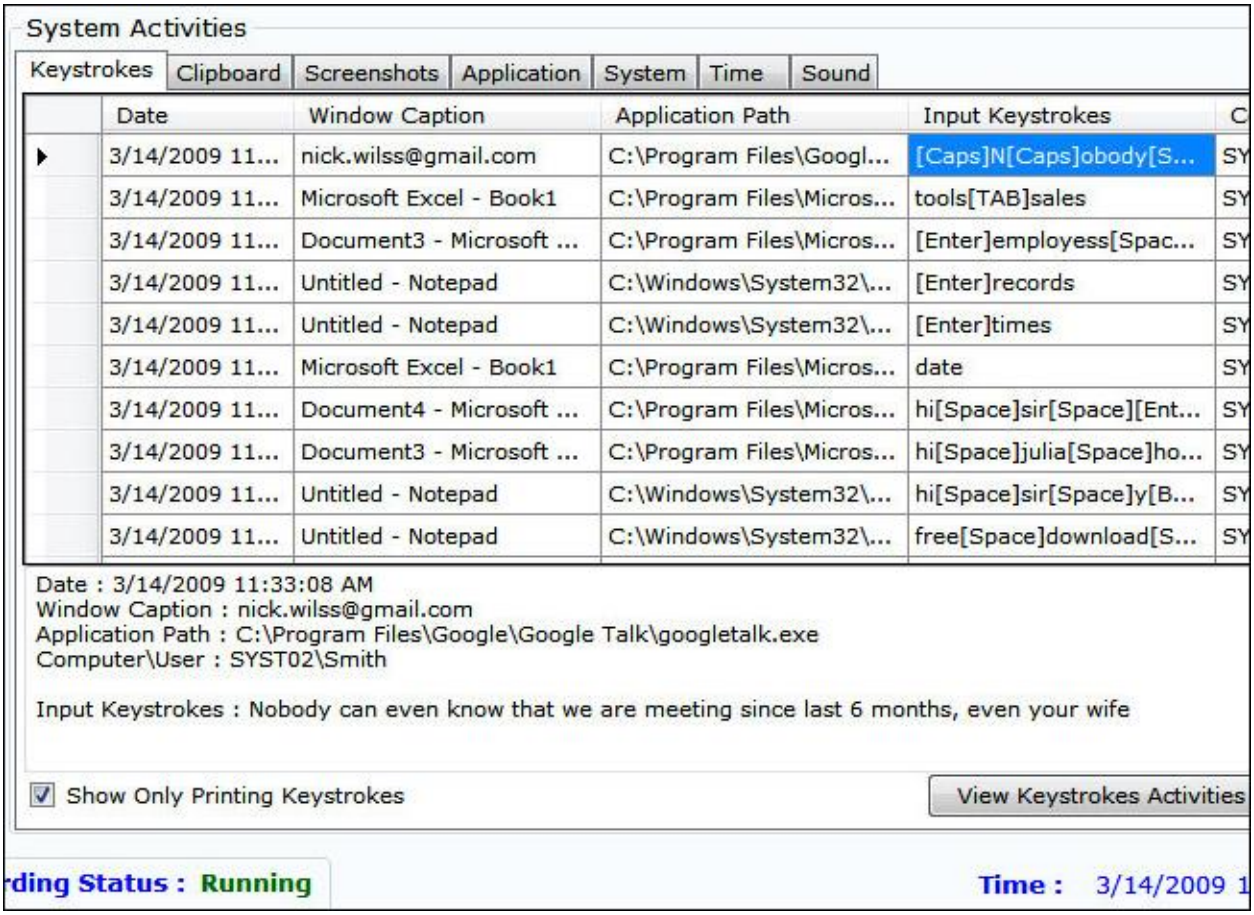
A keylogger is a piece of software — or, even scarier, a hardware device — that logs every key you press on your keyboard. It can capture personal messages, passwords, credit card numbers, and everything else you type. Keyloggers are generally installed by malware, but they may also be installed by protective parents, jealous spouses, or employers who want to monitor their employees. Hardware keyloggers are perfect for corporate espionage.

How a Keylogger Would Get On Your Computer

Most keyloggers on average computers arrive as malware. If your computer becomes compromised, the malware may include a keylogger or function as a Trojan that downloads the keylogger along with other harmful software. Keyloggers are a popular form of malware because they allow criminals to steal credit card numbers, passwords, and other sensitive data.

Keystroke-logging software may also be installed by someone close to you. A protective parent might go beyond typical parental controls and install software that includes a keylogger, allowing them to see everything their child types. A jealous spouse concerned about their husband or wife cheating might install a keylogger on their computer to keep tabs on them — it's not necessarily a good thing, but it happens.

Some employers might install keystroke loggers on their employees' computers to monitor everything they do, or just to investigate employees they're suspicious about. Laws vary about when this is legal from jurisdiction to jurisdiction.



Hardware Keyloggers

Some keyloggers can be implemented entirely as hardware devices. A typical desktop computer has a keyboard that connects to the back of the computer using a USB cable. If someone were to sneak in, unplug the keyboard's USB cable, then attach a specialized USB device between the computer's USB port and the keyboard's USB connector, the device could function as a keylogger. Sitting in the middle, it could intercept keyboard signals from the keyboard, store them on the device, and then pass the keystrokes to the computer so everything would appear to be working normally. Security software on the computer wouldn't be able to detect this keylogger, as it runs entirely in hardware. If the computer were hidden under a desk, no one would notice the device.

The person could then come back a few days later to grab the device and sneak off with it, leaving no trace of keylogging software or suspicious network activity.

If you're worried about hardware keyloggers, just check the back of your computer and ensure there's no suspicious device between your keyboard cable and the computer itself — of course, there probably won't be.

How Keyloggers Function

Keylogging software runs hidden in the background, making a note of each keystroke you type. Software could scan through the file for certain types of text — for example, it could look for sequences of numbers that look like credit card numbers and upload them to a malicious server so they can be abused.

Keylogging software may also be combined with other types of computer-monitoring software, so the attacker would be able to see what you typed when you visited your bank's website and narrow in on the information they want. A keylogger could detect the first keystrokes you typed into an online game or chat program, stealing your password.

Someone could also look through the entire log history to spy on you and see what you search for and type online. Computer-monitoring software intended for use by parents or employers may often combine the keylogger with a screenshot program, so someone can read through a history of what you typed combined with screenshots of what was on your computer screen at the time.

Form Grabbing

While key loggers typically record data for all programs on a system, form grabbers are specialized and only target data sent through a Web browser. When a user submits a Web form, such as those used to log onto a website, his Web browser generates an HTTP POST request that sends the data entered to the site. These data are normally encrypted using transport layer security (TLS) since it is very insecure to transmit logon and password data in cleartext. Form grabbers work by intercepting the POST data before the data pass through encryption routines.

Typosquatting

Typosquatting, also known as URL hijacking, is a form of cybersquatting (sitting on sites under someone else's brand or copyright) that targets Internet users who incorrectly type a website address into their web browser (e.g., "Goooogle.com" instead of "Google.com"). When users make such a typographical error, they may be led to an alternative website owned by a hacker that is usually designed for malicious purposes.

Hackers often create fake websites that imitate the look and feel of your intended destination so you may not realize you're at a different site. Sometimes these sites exist to sell products and services that are in direct competition with those sold at the website you had intended to visit, but most often they are intended to steal your personal identifiable information, including credit cards or passwords.

These sites are also dangerous because they could download malicious software to your device simply by visiting the site. So you don't even need to click on a link or accept a download for dangerous code to install on your computer, smartphone or tablet. This is called a drive-by download and many typosquatters employ this as a way to spread malicious software whose purpose is to steal your personal information.

In some cases, typosquatters employ phishing in order to get you to visit their fake websites. For example, when AnnualCreditReport.com was launched, dozens of similar domain names with intentional typos were purchased, which soon played host to fake websites designed to trick visitors. In cases like this, phishing emails sent by scammers spoofing a legitimate website with a typosquatted domain name make for tasty bait.

Spoofing

At the movies, a spoof is a comedy, masquerading as something else: An otherwise serious subject like a spy drama, crime caper, or action adventure, which is laced with self-mockery and comedic

elements. It's an effective strategy that pulls in fans of both comedy and the underlying genre that's being parodied – and one which generates profits and pleasure for millions.

In the cyber realm, imitation and misdirection serve a much darker purpose – and the payoff is usually designed to favor only a select few: The perpetrators.

What is Spoofing?

Spoofing in the digital sense occurs when a malicious outsider impersonates a legitimate resource like a website, email contact, network user, resource or device, with the intention of gaining access to systems and networks to stage attacks, steal valuable information, sabotage operations, or distribute malware. It's a popular tactic with cyber-criminals and takes several forms.

IP Address Spoofing

Digital networks communicate by exchanging data packets, each of which has multiple file headers for routing, and to ensure the continuity of their transmission. Among these is the Source IP Address header, which gives the IP (Internet Protocol) address that a data packet is sent from.

Also referred to as IP address forgery, IP spoofing or host file hijacking, IP address spoofing occurs when an attacker manages to obtain the IP address of a legitimate host then alters the Source IP Address headers on data packets sent from their own location, so that it appears that these packets originate from the legitimate source address that they've hijacked.

Having masked themselves as a trusted host, the attacker is free to impersonate a web site, gain access to networks to spy or launch attacks, or to hijack web browsers. If a user types a web address (URL or Uniform Resource Locator) into a hijacked browser, they may be misdirected to a spoofed web site designed to look just like the legitimate URL that the user typed in. There, they may unknowingly interact with malicious content concealed on the bogus page that could log their

key strokes (keylogging), act as a pipeline for attackers to steal or corrupt sensitive data, install malware, or take over infected systems.

And IP spoofing is the method of choice for launching Denial of Service or DoS attacks. Here, the target may be flooded with data packets from thousands of spoofed addresses, overloading their system. Alternatively, the IP address of the target may be spoofed and used to send data packets to multiple recipients – each of which will send packets back in return – in what's known as a reflected DDoS (Distributed Denial of Service) attack.

ARP Spoofing

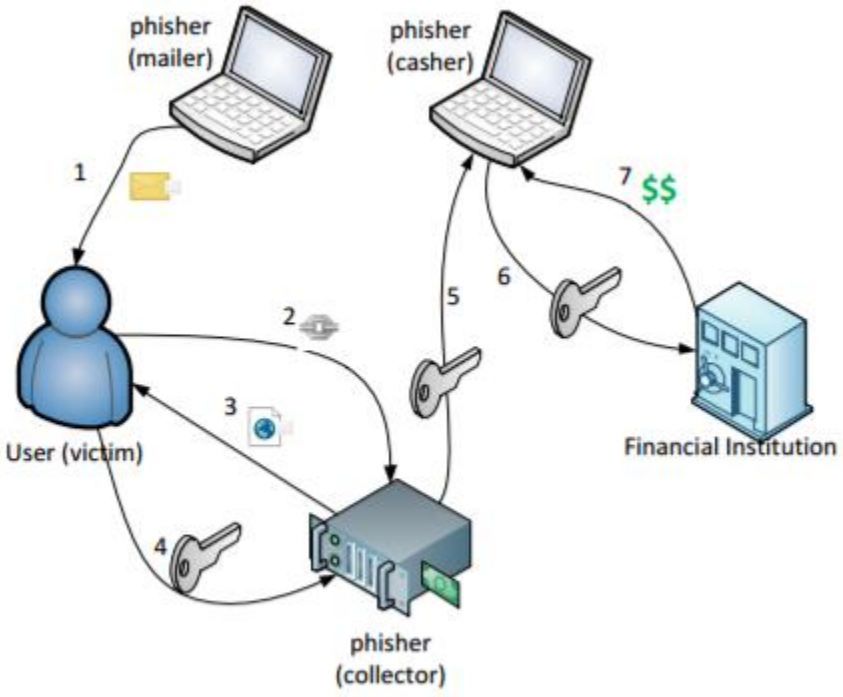
The Address Resolution Protocol (ARP) is used in resolving IP addresses with Media Access Control or MAC addresses for data transmission. ARP spoofing is done when an attacker transmits spoofed ARP communications across a local area network (LAN) so as to link their MAC address to the IP address of a legitimate network user. Any information intended for that user's IP address will be transmitted to the assailant, instead. For this reason, ARP spoofing is usually employed to steal data, modify it in transit, or for interfering with traffic on a LAN. The technique may also enable DoS and "man-in-the-middle" attacks and session hijacking.

DNS Server Spoofing

The Domain Name System or DNS resolves the URLs, email addresses and other text-based domain names that users type into their browsers with their associated IP addresses. In DNS server spoofing, attackers may compromise a DNS server and reroute a given domain name to a specific IP address – usually that of a server run by the attackers themselves. Users typing in that domain name will be redirected to a server that's typically laced with malware, viruses and worms.

Phishing

Phishing is a form of social engineering in which an attacker, also known as a phisher, attempts to fraudulently retrieve legitimate users' confidential or sensitive credentials by mimicking electronic communications from a trustworthy or public organization in an automated fashion. The word "phishing" appeared around 1995, when Internet scammers were using email lures to "fish" for passwords and financial information from the sea of Internet users; "ph" is a common hacker replacement of "f", which comes from the original form of hacking, "phreaking" on telephone switches during 1960s. Early phishers copied the code from the AOL website and crafted pages that looked like they were a part of AOL, and sent spoofed emails or instant messages with a link to this fake web page, asking potential victims to reveal their passwords.



A complete phishing attack involves three roles of phishers. Firstly, mailers send out a large number of fraudulent emails (usually through botnets), which direct users to fraudulent websites. Secondly, collectors set up fraudulent websites (usually hosted on compromised machines), which

actively prompt users to provide confidential information. Finally, cashers use the confidential information to achieve a pay-out. Monetary exchanges often occur between those phishers. The information flow is shown in Figure. Before delving further into phishing it's important to clarify what is not phishing. Nigerian 419 scam (sending emails to trick recipients into giving money to the scammer) and Internet auction fraud (non-delivery, misrepresentation, fee stacking, or selling stolen goods) are not considered phishing since they don't involve obtaining users' credentials. The latest statistics reveal that banks and financial institutions along with the social media and gaming sites continue to be the main focus of phishers. Some loyalty programs are also becoming popular among phishers because with them phishers can not only breech the financial information of victim but also use existing reward points as currency. U.S. remains the largest host of phishing, accounting for 43% of phishing sites reported in January 2012. Next was Germany at 6%, followed by Australia, Spain, Brazil, Canada, the U.K., France, Netherlands, and Russia. A study of demographic factors suggests that women are more susceptible to phishing than men and users between the ages of 18 and 25 are more susceptible to phishing than other age groups. Phishing attacks that initially target general consumers are now evolving to include high profile targets, aiming to steal intellectual property, corporate secrets, and sensitive information concerning national security

Types of Phishing

Phishing has spread beyond email to include VOIP, SMS, instant messaging, social networking sites, and even multiplayer games. Below are some major categories of phishing.

1. Clone Phishing

In this type phisher creates a cloned email. He does this by getting information such as content and recipient addresses from a legitimate email which was delivered previously, then he sends the same email with links replaced by malicious ones. He also employs address spoofing so that the email appears to be from the original sender. The email can claim to be a re-send of the original or an updated version as a trapping strategy.

2. Spear Phishing

Spear phishing targets at a specific group. So instead of casting out thousands of emails randomly, spear phishers target selected groups of people with something in common, for example people from the same organization. Spear phishing is also being used against high-level targets, in a type of attack called "whaling". For example, in 2008, several CEOs in the U.S. were sent a fake subpoena along with an attachment that would install malware when viewed. Victims of spear phishing attacks in late 2010 and early 2011 include the Australian Prime Minister's office, the Canadian government, the Epsilon mailing list service, HBGary Federal, and Oak Ridge National Laboratory.

3. Phone Phishing

This type of phishing refers to messages that claim to be from a bank asking users to dial a phone number regarding problems with their bank accounts. Traditional phone equipment has dedicated lines, so Voice over IP, being easy to manipulate, becomes a good choice for the phisher. Once the phone number, owned by the phisher and provided by a VoIP service, is dialed, voice prompts tell the caller to enter her account numbers and PIN. Caller ID spoofing, which is not prohibited by law, can be used along with this so that the call appears to be from a trusted source

Phishing vs Spoofing

Phishing is basically tricking someone to give up sensitive information – usually social and bank account credentials and credit card details. Spoofing, on the other hand, refers to how cybercrooks actually trick their target – by posing as a well-known, trustworthy entity.

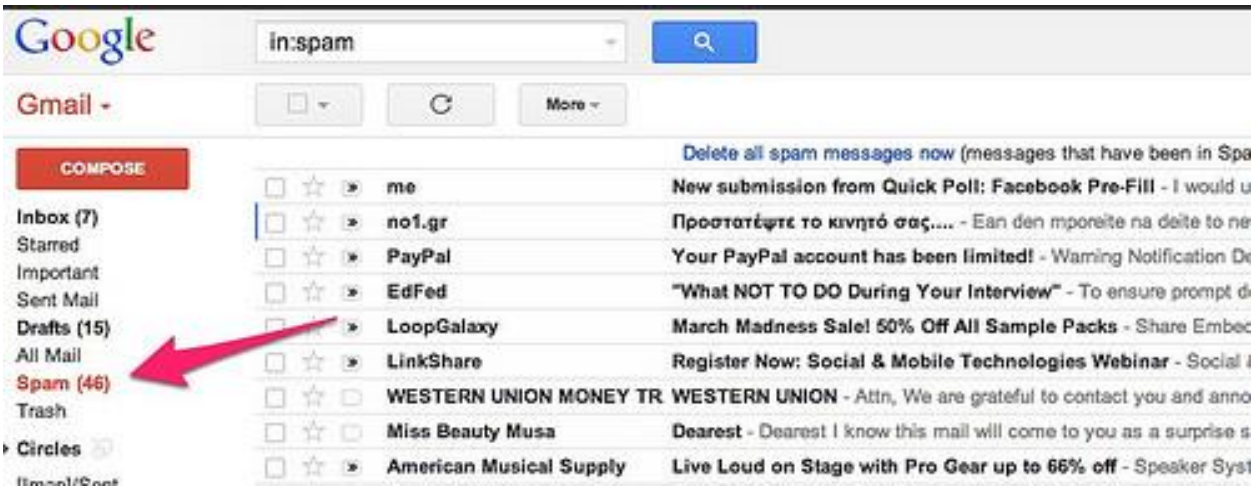
So, more often than not, phishers rely on spoofing in order for their phishing scams to be successful. For example, you receive an e-mail from your bank telling you your account has been suspended for whatever reason, and in order to reactivate it, you have to hand over your credit

card details. This, clearly, is a phishing attempt – your bank would never send you such e-mails! And the fact that the e-mail seems to come from your bank, when in fact it doesn't – that's spoofing.

Spamming

Electronic spamming is the use of electronic messaging systems to send an unsolicited message (spam), especially advertising, as well as sending messages repeatedly on the same site. While the most widely recognized form of spam is email spam, the term is applied to similar abuses in other media: instant messaging spam, Usenet newsgroup spam, Web search engine spam, spam in blogs, wiki spam, online classified ads spam, mobile phone messaging spam, Internet forum spam, junk fax transmissions, social spam, spam mobile apps, television advertising and file sharing spam. It is named after Spam, a luncheon meat, by way of a Monty Python sketch about a menu that includes Spam in every dish. The food is stereotypically disliked/unwanted, so the word came to be transferred by analogy.

Spamming remains economically viable because advertisers have no operating costs beyond the management of their mailing lists, servers, infrastructures, IP ranges, and domain names, and it is difficult to hold senders accountable for their mass mailings. Because the barrier to entry is so low, spammers are numerous, and the volume of unsolicited mail has become very high. In the year 2011, the estimated figure for spam messages is around seven trillion. The costs, such as lost productivity and fraud, are borne by the public and by Internet service providers, which have been forced to add extra capacity to cope with the deluge. Spamming has been the subject of legislation in many jurisdictions. A person who creates electronic spam is called a spammer.



Sniffing

Packet sniffing allows individuals to capture data as it is transmitted over a network and is used by network professionals to diagnose network issues, and by malicious users to capture unencrypted data, like passwords and usernames. If this information is captured in transit, a user can gain access to a system or network. If you want to keep information confidential or are concerned about packet sniffing, work on encrypted protocols and encrypt all sensitive data being sent over the Internet or network (e.g. e-mails). For example, e-mails can be encrypted using PGP, anyone still using Telnet should consider using SSH instead, and instead of FTP you can use SFTP.

Pharming

Pharming (pronounced 'farming') is a form of online fraud very similar to phishing as pharmers rely upon the same bogus websites and theft of confidential information. However, where phishing must entice a user to the website through 'bait' in the form of a phony email or link, pharming re-directs victims to the bogus site even if the victim has typed the correct web address. This is often applied to the websites of banks or e-commerce sites.

How pharming works

While there are several ways to pharm, the primary method stems from an older attack called DNS cache poisoning in which an attack is made against the Internet naming system that allows users to enter meaningful names for websites (such as www.bank.co.uk) rather than a series of numbers

(such as 192. 168. 1. 1.). The naming system relies upon DNS servers to handle the conversion of the letter-based website names, which are easily recalled by people into the machine-understandable digits that whisk users to the website of their choice. When a pharmer mounts a successful DNS cache poisoning attack, they are effectively changing the rules of how traffic flows for that portion of the Internet. It is from this practice that pharmers found their namesake – herding large numbers of Internet users to a bogus site rather than planting the 'bait' of the phishers.

Major instances of pharming

Due to the wide-scale effects pharming can have on large portions of the Internet, some pharming events have led to major new stories when used against a massive cooperation with thousands of visitors. Multiple prominent websites have been affected by pharming over the previous decade. A particularly newsworthy event occurred in 2004 when a German teenager hijacked the country's eBay domain name, leaving thousands of users redirected to a bogus site. Following in 2005, the domain name for Panix, a New York ISP, was redirected to a bogus site in Australia, while in the same year Hushmail, a secure email service, was attacked by redirecting users to a defaced website.

Man-in-the-Middle

A man-in-the-middle attack is a type of cyberattack where a malicious actor inserts him/herself into a conversation between two parties, impersonates both parties and gains access to information that the two parties were trying to send to each other. A man-in-the-middle attack allows a malicious actor to intercept, send and receive data meant for someone else, or not meant to be sent at all, without either outside party knowing until it is too late. Man-in-the-middle attacks can be abbreviated in many ways, including MITM, MitM, MiM or MIM.

Repudiate

Nonrepudiation is the assurance that someone cannot deny something. Typically, nonrepudiation refers to the ability to ensure that a party to a contract or a communication cannot deny the authenticity of their signature on a document or the sending of a message that they originated.

To repudiate means to deny. For many years, authorities have sought to make repudiation impossible in some situations. You might send registered mail, for example, so the recipient cannot deny that a letter was delivered. Similarly, a legal document typically requires witnesses to signing so that the person who signs cannot deny having done so.

On the Internet, a digital signature is used not only to ensure that a message or document has been electronically signed by the person that purported to sign the document, but also, since a digital signature can only be created by one person, to ensure that a person cannot later deny that they furnished the signature.

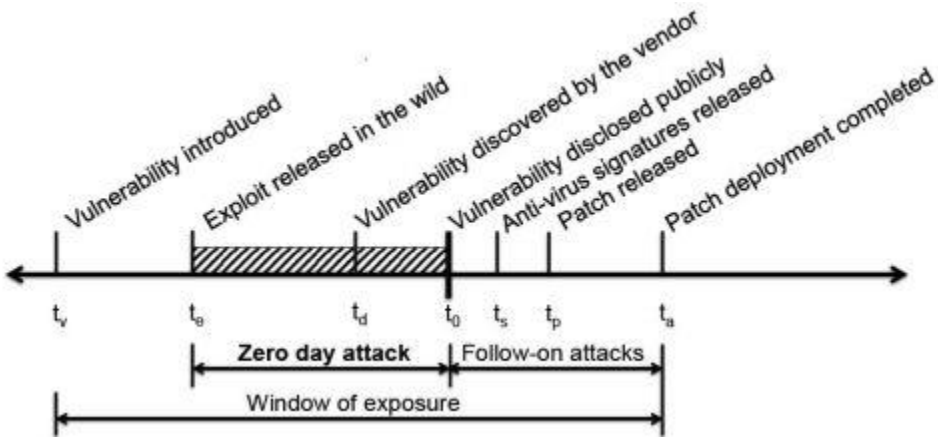
Since no security technology is absolutely fool-proof, some experts warn that a digital signature alone may not always guarantee nonrepudiation. It is suggested that multiple approaches be used, such as capturing unique biometric information and other data about the sender or signer that collectively would be difficult to repudiate. Email nonrepudiation involves methods such as email tracking that are designed to ensure that the sender cannot deny having sent a message and/or that the recipient cannot deny having received it.

Zero Day Vulnerability

A zero-day vulnerability refers to a hole in software that is unknown to the vendor. This security hole is then exploited by hackers before the vendor becomes aware and hurries to fix it—this exploit is called a zero day attack. Uses of zero day attacks can include infiltrating malware, spyware or allowing unwanted access to user information. The term “zero day” refers to the unknown nature of the hole to those outside of the hackers, specifically, the developers. Once the vulnerability becomes known, a race begins for the developer, who must protect users. In order for the vendor to rectify the vulnerability, the software company must release a patch. Often patches are released on a regular basis, one example being Microsoft’s Patch Tuesday. On the second Tuesday of each month, Microsoft releases security fixes that resolve identified holes. If, however, a critical vulnerability is discovered, a patch may be released outside of schedule.

Browsers are similarly vulnerable; it's a good idea to update your browser often, for updated security as well as features. To check if any updates are available for your browser of choice, open the browser and click either "Help" or the browser name, depending on which browser you're using. A quick online search will provide step-by-step instructions. Alternately, you could set up automatic updates, again, depending on browser.

Zero day vulnerabilities can be serious security risks. When searching for an appropriate antivirus solution, look for security software that protects against both known and unknown threats.



Ransomware

Ransomware is a type of malware that restricts access to the infected computer system in some way, and demands that the user pay a ransom to the malware operators to remove the restriction. There are two main forms of ransomware in circulation today:

Locker ransomware (computer locker): Denies access to the computer or device

Crypto ransomware (data locker): Prevents access to files or data.

Crypto ransomware doesn't necessarily have to use encryption to stop users from accessing their data, but the vast majority of it does. Both types of ransomware are aimed squarely at our digital lifestyle. They are designed to deny us access to something we want or need and offer to return

what is rightfully ours on payment of a ransom. Despite having similar objectives, the approaches taken by each type of ransomware are quite different.

Locker ransomware (Computer locker)

Locker ransomware is designed to deny access to computing resources. This typically takes the form of locking the computer's or device's user interface and then asking the user to pay a fee in order to restore access to it. Locked computers will often be left with limited capabilities, such as only allowing the user to interact with the ransomware and pay the ransom. This means access to the mouse might be disabled and the keyboard functionality might be limited to numeric keys, allowing the victim to only type numbers to indicate the payment code. Locker ransomware is typically only designed to prevent access to the computer interface, largely leaving the underlying system and files untouched. This means that the malware could potentially be removed to restore a computer to something close to its original state. This makes locker ransomware less effective at extracting ransom payments compared with its more destructive relative crypto ransomware. Techsavvy victims are often able to restore access using various tools and techniques offered by security vendors. Because locker ransomware can usually be removed cleanly, it tends to be the type of ransomware that goes to great lengths to incorporate social-engineering techniques to pressure victims into paying. This type of ransomware often masquerades as law enforcement authorities and claims to issue fines to users for alleged online indiscretions or criminal activities. Locker ransomware can particularly be effective on devices that have limited options for users to interact with. This is a potential problem area considering the recent boom in wearable devices and the Internet of Things (IoT), where millions of connected devices could potentially be at risk from this type of ransomware.

Crypto ransomware (Data locker)

This type of ransomware is designed to find and encrypt valuable data stored on the computer, making the data useless unless the user obtains the decryption key. As people's lives become increasingly digital, they are storing more important data on their personal computers and devices. Many users are not aware of the need to create backups to guard against hard disk failures or the

loss or theft of the computer, let alone a possible crypto ransomware attack. This could be because users don't have the knowhow or don't realize the value of the data until it is lost. Setting up an effective backup process requires some work and discipline, so it's not an attractive proposition for the average user. Crypto ransomware targets these weaknesses in the typical user's security posture for extortion purposes. The creators of crypto ransomware know that data stored on personal computers is likely to be important to users. For example, the data could include things like memories of loved ones, a college project due for submission, or perhaps a financial report for work. The ransomware victims may be desperate to get their data back, preferring to pay the ransom to restore access rather than simply lose it forever and suffer the consequences. After installation, a typical crypto ransomware threat quietly searches for and encrypts files. Its goal is to stay below the radar until it can find and encrypt all of the files that could be of value to the user. By the time the victim is presented with the malware's message that informs them that their data is encrypted, the damage is already done. With most crypto ransomware infections, the affected computer continues to work normally, as the malware does not target critical system files or deny access to the computer's functionality. This means that users can still use the computer to perform a range of activities apart from accessing the data that has been encrypted.

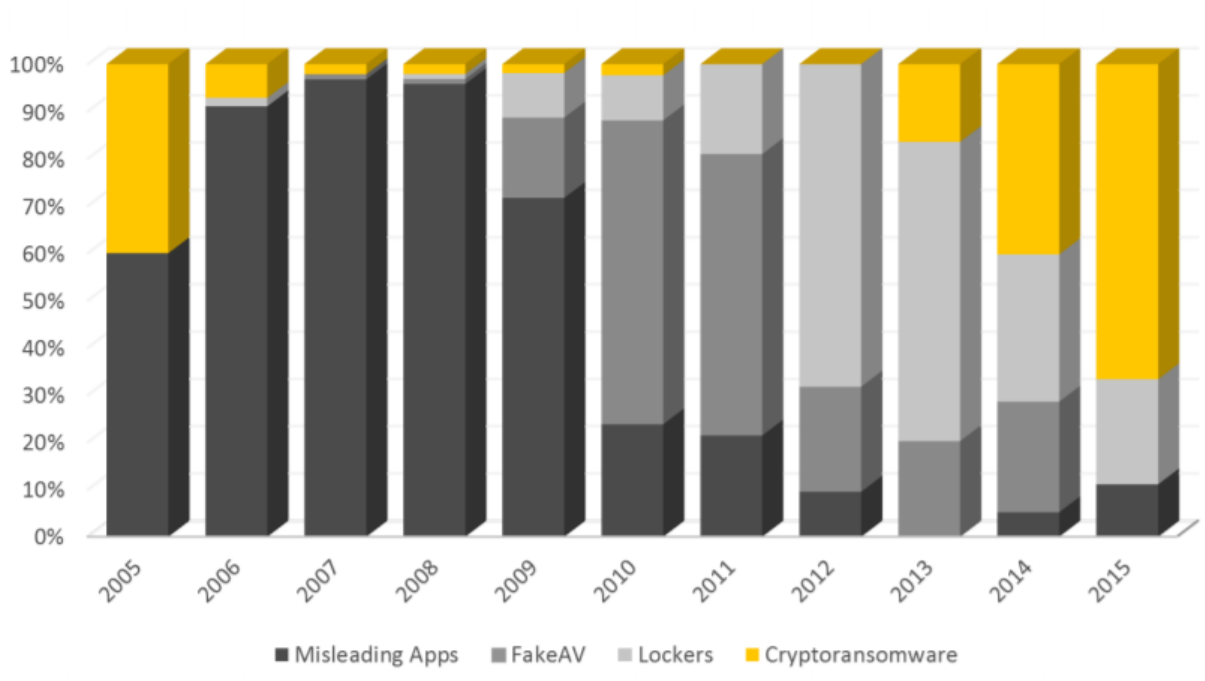
How ransomware has evolved

The evolution of ransomware has been greatly influenced by a range of developments in technology, economics, security, and culture since 1989. Today's ransomware is a sophisticated threat affecting users in many regions worldwide, particularly those living in developed and high-tech economies. The ransomware world is like any real life ecosystem. Threats that can adapt and evolve to their surroundings can survive and even thrive, while those that can't or won't adapt may eventually disappear. The ransomware world is a good example of where Darwinian-style evolution is at work. Ransomware origins. The modern-day ransomware has evolved considerably since its origins 26 years ago with the appearance of the AIDS Trojan. The AIDS Trojan was released into the unsuspecting world through snail mail using 5¼" floppy disks in 1989. Despite the public being unprepared for this new type of threat all those years ago, the AIDS Trojan was ultimately unsuccessful due to a number of factors. Back then, few people used personal computers, the

World Wide Web was just an idea, and the internet was mostly used by experts in the field of science and technology. The availability and strength of encryption technology was also somewhat limited at the time. Along with this, international payments were harder to process than they are today. While the emergence of the AIDS Trojan established the ransomware threat, this type of malware didn't get widely used in cybercrime until many years later. The threat landscape was considerably different back in the nineties and early noughties. That was an era when malware was used in pranks and vandalism to gain notoriety; nowadays, malware is mostly being deployed for financial gain. The evolution of ransomware, particularly crypto ransomware, accelerated in recent years as more copycat criminal enterprises jumped into the arena to build on others' success.

Pivotal moments in ransomware history

As we look at the recent history of ransomware, it is useful to consider the overall picture of money payment/ extortion threats over the past 10 years to get an idea of where modern-day ransomware evolved from. The graph in Figure shows how the market for extortion malware has been divided up each year since 2005. While each threat never disappeared entirely, it's easy to identify how preferences shifted from one type of extortion malware to another.



Misleading applications and early ransomware

The first wave of misleading applications began to appear in 2005. The apps posed as fake spyware removal tools, such as SpySherriff, or performance enhancement tools, such as PerformanceOptimizer and RegistryCare. These fake tools mainly affected Windows computers, but also targeted Mac OS X computers. They typically exaggerated the impact of issues on the computer, such as unused registry entries and corrupt files, and said that they would resolve these issues if the user paid between US$30 and US$90 for a license. In reality, many of them did not fix anything.

Even at this early stage, the first wave of modern crypto ransomware threats appeared. The Trojan.Gpcoder family emerged in May 2005, initially using custom-encryption techniques which were weak and easily overcome. They also used symmetric encryption algorithms, which meant the same key was used for both encryption and decryption. Despite initial failures, the malware authors did not give up and continued to create newer versions of the threat, making refinements at each step as they learned the lessons from the past failures. By early 2006, the concept of crypto ransomware started to gain traction as attackers started to experiment with the idea. This renaissance in crypto ransomware led to the appearance of threats like Trojan.Cryzip in March 2006. Cryzip copied data files into individual password-protected archive files and then deleted the originals. However, the password was actually embedded inside the code of the Trojan itself, making it easy to recover the password. Trojan.Archiveus also emerged in 2006. Like Cryzip, Archiveus used password-protected archive files but in a bizarre twist, the malware did not ask for cash payment. Instead, it asked the victim to buy medication over the internet using certain online pharmacy URLs. The victim then needed to submit the order ID to get the password to decrypt the archive files. In this way, the attackers could have earned commission from the purchase which was then considered as a ransom payment–though the makers of Archiveus would not have approved of this terminology

Fake AV

The next pivot point happened between 2008 and 2009, when cybercriminals switched to using fake antivirus programs, a more aggressive subcategory of misleading applications. The tools mimicked the appearance and functionality of legitimate security software and performed mock scans, claiming to find large numbers of threats and security issues on the computer. The user was then asked to pay a fee of between US$40 and US$100 to fix the fake problems. They may also have been asked to pay for bogus multi-year support services. However, some fake AV victims chose to ignore the alerts or removed the software, resulting in a lower return on investments for the cybercriminals.

The move to locker ransomware

From 2011 to 2012, attackers transitioned from fake antivirus tools to a more disruptive form of extortion. This time, the cybercriminals disabled access and control of the computer, effectively locking up the computer from use. In terms of ransom amounts, locker ransomware pushed up the benchmark compared with fake antivirus and misleading apps. A typical locker ransomware threat charges around US$150 to US$200 payable through electronic cash vouchers.

Locker ransomware emerged a few years before its peak between 2011 and 2012. The first of the pure computer-locking malware hit users around the start of 2008 in the shape of Trojan.Randsom.C. This pioneer spoofed a Windows Security Center message and asked the user to call a premium-rate phone number to reactivate a license for security software. The computer was locked during this time, so the user was unable to use the computer for any other purpose.

As locker ransomware was refined, it went from just reporting non-existent errors to actually beginning to introduce errors and problems. Eventually, it dropped any pretense of being a helpful tool to just displaying a blatant request for payment to restore access to the computer. This is because in the early days, attackers tricked victims into downloading fake tools to fix computer issues. Today, ransomware can be installed without any user interaction through attacks such as drive-by downloads. Despite this, locker ransomware creators still continued to use socialengineering techniques to convince users to pay the ransom. The threats began to pose as law enforcement notices instead of antivirus software and system performance tools. They typically claimed that the user had broken the law by downloading copyrighted materials such as pirated music, movies, or software or viewing other illegal digital materials such pornographic images depicting minors or animals.

The move to crypto ransomware

Deficiencies in all the other extortion schemes ultimately led the cybercriminals back to the original type of ransomware. From 2013 to the present day, there has been a pivot back to crypto ransomware. Crypto ransomware tends not to use social engineering; instead it is upfront about its intentions and demands. The threats typically display an extortion message, offering to return data upon payment of hefty ransoms. Crypto ransomware has raised the ransom amounts bar to a new level. A typical crypto ransomware threat requests payment of around US$300 for a single

computer. Today's crypto ransomware threats are much more capable than its predecessors, with stronger operational and encryption procedures

Targets for ransomware

- Home users

Ransomware is perhaps the most effective against individuals who are not fluent with computers or are not familiar with ransomware and how it works. The most common group that we see impacted by ransomware is the home user, who often has the least amount of access to technical assistance. The lack of support may leave the user feeling isolated and helpless, further increasing the pressure to pay. Home users often have sensitive information, files, and documents that are personally valuable stored on the computer, such as college projects, photos, and video game save files. Despite these things being of value to users, home users are still unlikely to have an effective back up strategy in place to successfully recover from events such as a fire or theft, let alone a crypto ransomware attack. A previous survey by Symantec/Norton showed that 25 percent of home users did not do any backups at all. Fifty-five percent backed up some files. In terms of backup frequency, only 25 percent of users backed up files once a week. The rest only made backups once a month or even less frequently than that. This means users are potentially leaving themselves exposed in the event of a ransomware attack. Even if the home user has a backup process, some threats delete local backups on the computer and encrypt backup files on external storage devices that are connected to the computer.

- Businesses

For many businesses, information and the technology to use it is their life blood, without which the act of conducting day-to-day business is impossible. Consider a retailer running a computerized point-of-sale (POS) system. If the POS system was unavailable due to a ransomware infection, the retailer would not be in a position to transact sales. Business computers are also more likely to contain sensitive data and documents of critical importance, such as customer databases, business plans, proposals, reports, source code, forms, and tax compliance documents. Modern crypto ransomware threats can enumerate all accessible drives, such as local file-share servers, and encrypt files on these as well. This means more than one system can be impacted by just a single crypto ransomware infection. The loss of this information could have a catastrophic impact on the business. While many companies have backup and disaster recovery plans, there are still many who do not. Some organization's disaster recovery plans may not

extend to cover the individual end users. Even if the businesses had plans, it is quite possible that they have not been tested and may not work as expected when required. These factors make individual business users a viable target for traditional crypto ransomware. Aside from ransomware impacting individual business users, there have also been cases reported where the company itself had been targeted with file-encrypting ransomware. In a case involving PHP.Ransomcrypt.A, the attackers were believed to have compromised an organization for months, quietly encrypting the database along with all of the incremental backups. At the appropriate time, the attackers made their substantial ransom demands known to the business, threatening them with the potential loss of several months' worth of data.

Public agencies

Public agencies such as educational institutes and even law enforcement entities are not excluded from the attention of these cybercriminals and in some cases, they may be specifically targeted. There have been several reports of law enforcement agencies that had been hit with crypto ransomware in the past. In another case, a New Jersey school district, which runs four elementary schools in the Swedesboro-Woolwich area, was hit by cybercriminals who demanded a ransom payment of 500 bitcoins (US$124,000). The latter incident proved to be highly disruptive, as the attackers compromised computers and files used by staff and students. These cases highlight the brazenness of the attackers who are not even afraid of holding law enforcers to ransom. The cybercriminals believe that they are beyond the reach of the law by operating from another jurisdiction.

Propagation

One of the first questions many victims ask is "how did I get infected with ransomware?" While it is not always immediately clear, the infection method for ransomware follows the same modus operandi used by cybercriminals to infect victims with any malware. As seen in Figure 10, there are many paths that can lead to a ransomware infection. However, the skillset and resources required to overcome modern defenses for the distribution of malware is outside of the scope of many amateur cybercriminals. This has led to an underground cybercrime ecosystem where different groups specialize in distinct areas of cybercrime, such as malware distribution, for a price. In many ways, these malware distribution services are run like any other business service. In some cases, they have even adopted common software industry compensation methods for malware

installs, such as the pay-per-install (PPI) model. Ransomware attackers have been seen to use different techniques or services to get their malware onto a victim's computer.

Traffic distribution system (TDS)

A common method used by these distribution services is to buy redirected web traffic from a Traffic Distribution Service (TDS) vendor and point it to a site hosting an exploit kit. In a lot of cases, the redirected traffic originates from adult content-related websites. If the exploit kit is successful in exploiting a vulnerability in the visiting victims' computer, it can lead to what is commonly referred to as the drive-by-download of malware.

Malvertisement

Similarly, malicious advertisements known as malvertisments can get pushed onto legitimate websites in order to redirect traffic to a site hosting an exploit kit. In one case, we even observed unintentional cross contamination as a result of a click-fraud malware infection, where clicking on the malvertisment led to a ransomware infection. In both cases, cybercriminals can use real-time bidding to purchase traffic or ad space of interest that can allow them to geographically target victims and operate without borders.

Spam email

For many years, email spam using social-engineering themes has been the method of choice for distributing all types of malware including ransomware. Cybercriminals use a botnet to send the spam. These cybercriminals may also offer a spamming service to other attackers for a fee. The spam usually comes in the form of an email containing a malicious attachment or a link in the email leading to a site hosting an exploit kit. The spam may also involve the download of malware through other social-engineering means. The spam emails embody a whole range of socialengineering and psychological levers to trick users into installing the ransomware.

Downloaders & botnets

This method is one of a number of ways to distribute malware known as downloaders. Once the downloader infects a computer, its job is to download secondary malware onto the compromised

system. The cybercriminals behind downloaders offer a malware-installation service onto already compromised computers, at a price to other malware authors. Trojan botnets have also been known to download ransomware onto computers they have infected. This is usually done by cybercriminals as a final way of monetizing infected computers that they control.

Social engineering and self-propagation

Some ransomware also contain functionality to spread. For example, on Android, there are some samples that not only lock the device or encrypt files, but employ worm-like capabilities to spread to all contacts within the device's address book by sending social-engineering SMS messages. On the Windows platform, a variant of the Ransomlock (W32.Ransomlock.AO) screen locker is known to infect other files as a way to spread. Self-propagation is potentially an effective way for the ransomware to spread itself, but it does cause problems for the cybercriminals who are hoping for a ransom to be paid. If the ransomware is continuously spreading through a network, infecting multiple computers and demanding payment each time, the cybercriminal's promise to repair the damage after the victim pays the ransom is broken. Nobody will be willing to pay if the same gang continues to demand ransom payment after payment.

Ransom techniques

While all ransomware types are designed to extort money from their victims, they can be quite different both operationally and technically. To understand just how different they can be, this section will look at common locker ransomware and crypto ransomware to see how they work on a technical level.

- File encryption

Modern crypto ransomware typically uses both symmetric and asymmetric encryption techniques. In symmetric encryption, a single key is used to encrypt the data and the same key is used to decrypt the encrypted data. Knowing the key allows the user to decrypt data that has been encrypted with the same key. Ransomware using symmetric encryption will usually generate a key on the infected computer and send this to the attacker or request a key from the attacker before

encrypting the user's files. The attacker needs to ensure that the key is not available to the user after encrypting their files, otherwise the user might be able to decrypt the files themselves without paying. The advantage of using symmetric encryption algorithms is that they are generally much faster than asymmetric algorithms and use small keys (typically 256-bit). A typical crypto ransomware has to quickly search and encrypt a large number of files, so performance is essential to encrypt files before the victim can discover the threat's activities. Asymmetric encryption uses two keys: the public key is used to encrypt the data and the private key is used to decrypt the encrypted data. Knowing the public key does not allow you to decrypt files encrypted with this key. Only the related private key can be used for this purpose. Crypto ransomware may use asymmetric encryption by encrypting the user's files with the public key with the attacker keeping the private key for themselves. The attacker does not need to be as protective of the public key as they would need to be with the symmetricencryption approach, because knowing the public key does not allow the affected user to decrypt their files. There are a number of drawbacks to using a public key to encrypt huge numbers of potentially large files. Public key cryptography is much slower than symmetric key encryption. Taking a long time to complete encryption could risk exposing the operation before the encryption process is fully completed. More advanced crypto ransomware typically uses a combination of symmetric and asymmetric encryption techniques. The variants that use asymmetric encryption may also generate specific public-private key pairs for each infected computer. This allows the attacker to decrypt files on one infected computer without revealing the private key that could potentially also be used to decrypt files on every other computer infected using the same pubic key. The location of the keys in either encryption approach can have a fundamental impact on the effectiveness of the scheme and ultimately the outcome for the user. For example, if a key is generated on the infected computer and then sent to the attacker, then the user's files can be encrypted even if the crypto ransomware cannot contact the attacker's server. If the encryption key is only stored on the attacker's server, then the file-encryption process cannot begin unless the ransomware can contact the server and download the encryption key. A fundamental weakness in this approach is its dependency on a remote server before the start of operation.

- Downloaded public key

Cryptodefense (Trojan.Cryptodefense) uses a combination of symmetric and asymmetric encryption techniques. AES is a powerful and fast symmetric encryption algorithm which is used by Cryptodefense to encrypt the user's files. The 256-bit AES key is first generated on the user's computer and after file encryption is completed, the AES key is itself encrypted with a different RSA asymmetric public key which is downloaded from the attacker's server. The resulting encrypted AES key is then stored in the user's encrypted file. Even though the AES key is stored in each encrypted file on the user's computer, the victim has no way of using it as the attacker controls the RSA private key needed to decrypt it. The weakness of this approach is that if the attacker's server cannot be reached to download the RSA public key, then the encryption process will not be successful. The advantage of this approach is that the attacker can use a different RSA asymmetric key pair for each infection. Exposure of a single RSA private key will not allow any other victims to unlock their files.

- Embedded public key

To build trust, some crypto ransomware schemes allow the victim to "try-before-you-buy" by decrypting some files for free. For example, CTBLocker (Trojan.Cryptolocker.G) has an option to allow users to decrypt five randomly chosen files for free. This is a trust-building exercise to show victims that the cybercriminals can and are willing to decrypt files–if the ransom is paid. CTBLocker also uses both symmetric and asymmetric encryption techniques to encrypt the user's files but takes a slightly different approach. Samples of CTBLocker include an embedded public key for the RSA asymmetric encryption algorithm process. The attacker keeps the corresponding private key. During the infection process, CTBLocker generates a new symmetric key for the AES encryption process and uses it to encrypt the user's files. The 256-bit AES key is encrypted with the embedded public RSA key and the encrypted AES key is then added to the encrypted file's data. The user cannot recover the AES key to decrypt their files as they do not possess the private RSA key needed to decrypt the key. The advantage of using this approach is that CTBLocker can begin its file-encryption process without requiring any internet access first. The weakness of using this

approach is that attackers must use a different public key for each infection of CTBLocker. If they don't do this, then once the first user obtains the private RSA key, they could potentially share the key with other victims, allowing them to decrypt their files. For this scheme to be effective, the attacker must customize each copy of CTBLocker sent to victims.

- Embedded symmetric key

In 2014, we also saw the emergence of crypto ransomware for Android devices in the shape of Android.Simplocker. Android.Simplocker only uses the AES symmetric encryption algorithm to encrypt files on the user's mobile device. The 256-bit AES key is included in the application code itself so the malware does not need to reach out to a C&C server to download any additional keys or files. Instead, the attacker can instruct Simplocker by sending a command to it through an SMS message, for example, to direct the ransomware to encrypt or decrypt the user's files. As the key is included in the application, it is relatively straight forward to find the key and use it to decrypt the encrypted files. Hard coding symmetric encryption keys in this way is not a common technique for modern crypto ransomware. The method is usually only seen in the most basic forms of crypto ransomware such as those from amateur newcomers who have not learned past lessons on cryptography.

Screen locking

Locker ransomware attempts to block infected users from accessing the operating system and services that are running on their computer or device. The approach that is most commonly used is to display a ransom message to the user in a continuous loop. This gives the impression that the message is constantly displayed even though there may be slight intervals where it is possible for the user to close the current display of the message. These ransomware threats mostly use features or APIs from the underlying operating system to perform this task.

Windows locker ransomware

The locker ransomware threats that infect the Windows operating system, such as Trojan. Ransomlock.G, all employ similar strategies to lock the user's screen. The ransomware displays a

full screen window that covers the entire desktop to display its message. The ransomware may create the window itself or use a browser window in full screen mode to show their ransom message. The window is usually shown as the only window on a new virtual desktop that the ransomware creates and makes active. The ransomware may use a background thread to monitor the system's desktops and ensure that their one is kept active and on top. The contents of the messages are occasionally included in the ransomware executable itself but it is more common for the ransomware to download the contents from the attackers' server. This allows the attackers to serve localized messages using language and law-enforcement images relevant to the country where the infection has occurred. For self-protection, locker ransomware on Windows often use background threads to monitor for processes and applications that the user may try to use to end the ransomware process, such as Task Manager. The ransomware process will end these processes if they are detected. Some variants have also used shutdown messages to try to signal to other windows that the system is shutting down. This may allow the ransomware to close other processes that may interfere with its activities.

Denial of service

When you type a URL for a particular website into your browser, you are sending a request to that site's computer server to view the page. The server can only process a certain number of requests at once, so if an attacker overloads the server with requests, it can't process your request. This is a "denial of service" because you can't access that site.

In computing, a denial-of-service attack (DoS attack) is a cyber-attack where the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled. A DoS attack is analogous to a group of people crowding the entry door or gate to a shop or business, and not letting legitimate parties enter into the shop or business, disrupting normal operations.

Criminal perpetrators of DoS attacks often target sites or services hosted on high-profile web servers such as banks or credit card payment gateways. Revenge, blackmail and activism can motivate these attacks.

Denial-of-service attacks are characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service. In a DDoS attack, the incoming traffic flooding the victim originates from many different sources – potentially hundreds of thousands or more. This effectively makes it impossible to stop the attack simply by blocking a single IP address; plus, it is very difficult to distinguish legitimate user traffic from attack traffic when spread across so many points of origin. There are two general forms of DoS attacks: those that crash services and those that flood services. The most serious attacks are distributed. Many attacks involve forging of IP sender addresses (IP address spoofing) so that the location of the attacking machines cannot easily be identified and so that the attack cannot be easily defeated using ingress filtering.

How does an attack work?

One way to attack a company's network or website is to flood its systems with information. Web and e-mail servers can only handle a finite amount of traffic and an attacker overloads the targeted system with packets of data. Denial-of service attacks can essentially disable the computer or the network. Depending on the nature of the enterprise, this can disable your organization. Some denial-of-service attacks can be executed with limited resources against a large, sophisticated site. This type of attack is sometimes called an "asymmetric attack". For example, an attacker with an old PC and a slow modem may be able to disable much faster and more sophisticated machines or network

Mafia Boy Attack

Michael Calce (born 1986, also known as MafiaBoy) was a high school student from West Island, Quebec, who launched a series of highly publicized denial-of-service attacks in February 2000 against large commercial websites, including Yahoo!, Fifa.com, Amazon.com, Dell, Inc., E*TRADE, eBay, and CNN. He also launched a series of failed simultaneous attacks against 9 of the 13 root name servers

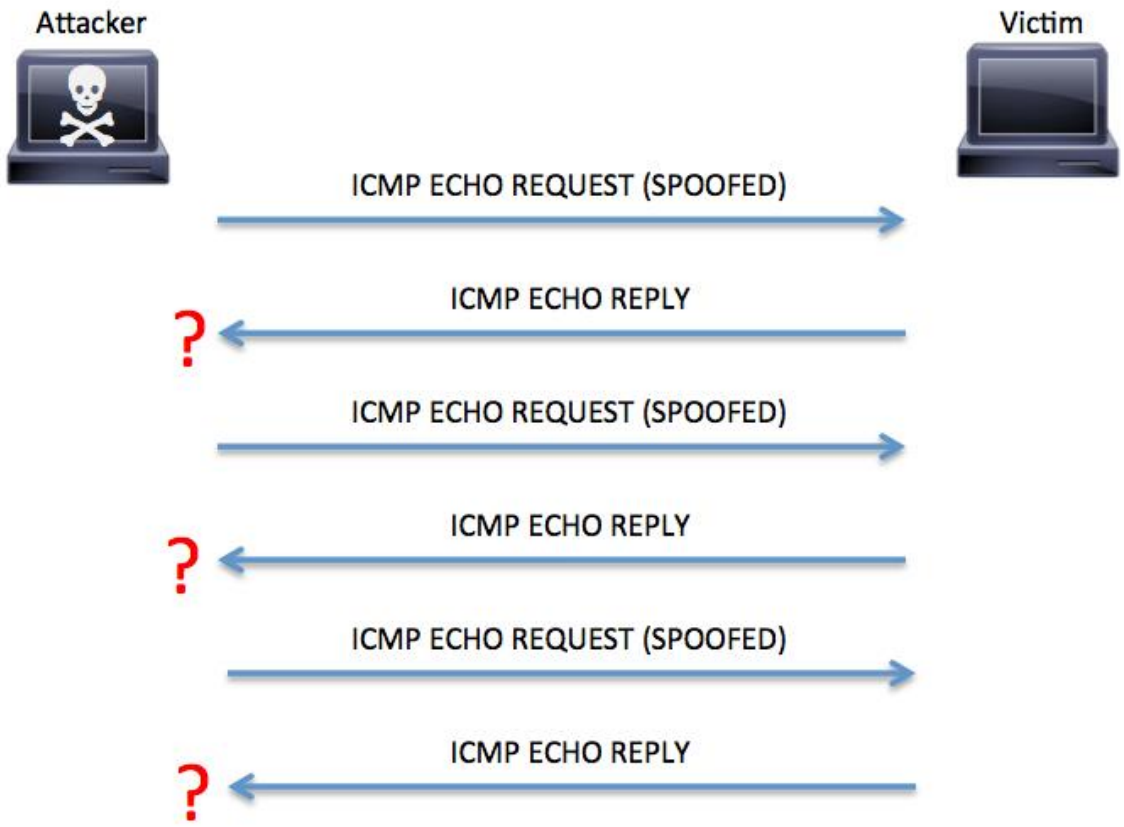
On February 7, 2000, Calce targeted Yahoo! with a project he named Rivolta, meaning “riot” in Italian. Rivolta was a denial-of-service attack in which servers become overloaded with different types of communications to the point where they shut down completely. At the time, Yahoo! was a multibillion-dollar web company and the top search engine. Mafiaboy's Rivolta managed to shut down Yahoo! for almost an hour. Calce's goal was, according to him, to establish dominance for himself and TNT, his cybergroup, in the cyberworld. Buy.com was shut down in response. Calce responded to this in turn by bringing down eBay, CNN, Amazon and Dell.com via DDoS over the next week.

In a 2011 interview, Calce claimed that the attacks had been launched unwittingly, after inputting known addresses in a security tool he had downloaded from a repository on the now defunct file-sharing platform Hotline, developed by Hotline Communications. Calce would then have left for school, forgetting the application which continued the attacks during most of the day. Upon coming home Calce says that he found his computer crashed, and restarted it unaware of what had gone on during the day. Calce claimed when he overheard the news and recognized the companies mentioned being those he had inputted earlier in the day, that he had "started to understand what might have happened".

Ping Flood Attack

One of the oldest network attacks around. It's goal is to saturate the network with ICMP traffic. No very effective today, because it requires a large amount of bandwidth to be successful. However, a small variation of the attack method can make it still feasible.

Ping, a well-known command line utility, is used to send ICMP packets to test whether a particular host is reachable in the network. In Ping flood attacks, these ICMP packets are exploited. This is done by sending a large number of ping ‘echo requests’ from a large number of ‘zombies’ which spread across the network. The victim will respond with ICMP Echo Reply packets, thus consuming both outgoing bandwidth as well as incoming bandwidth and it cripples the network such that the target victim is unable to respond to any other legitimate requests as it needs to process this massive number of ‘echo requests’.



The above image shows an attacker utilizing the advantage of ICMP's basic IP Connectivity test. Here, un-fragmented ICMP ECHO REQUEST packets are sent to the targets server until it's stopped. The victim responds to each of these requests with ICMP ECHO REPLY thus limiting the amount of available system resources for other processes. These continuous requests and replies can slow the network down by consuming both outgoing and incoming bandwidth thus causing legitimate traffic to continue at a significantly slow speed or sometimes can be brought to a complete halt.

Smurf Attack

Smurf is a network layer distributed denial of service (DDoS) attack, named after the DDoS.Smurf malware that enables it execution. Smurf attacks are somewhat similar to ping floods, as both are carried out by sending a slews of ICMP Echo request packets. Unlike the regular ping flood, however, Smurf is an amplification attack vector that boosts its damage potential by exploiting characteristics of broadcast networks.

The steps in a Smurf attack are as follows:

- First, the malware creates a network packet attached to a false IP address — a technique known as "spoofing."
- Inside the packet is an ICMP ping message, asking network nodes that receive the packet to send back a reply
- These replies, or "echoes," are then sent back to network IP addresses again, setting up an infinite loop.

When combined with IP broadcasting — which sends the malicious packet to every IP address in a network — the Smurf attack can quickly cause a complete denial of service.

Ping of Death

Ping of Death (a.k.a. PoD) is a type of Denial of Service (DoS) attack in which an attacker attempts to crash, destabilize, or freeze the targeted computer or service by sending malformed or oversized packets using a simple ping command.

While PoD attacks exploit legacy weaknesses, which may have been patched in target systems. However, in an unpatched system, the attack is still relevant and dangerous. Recently, a new type of PoD attack has become popular. This attack, commonly known as a Ping flood, the targeted system is hit with ICMP packets sent rapidly via ping without waiting for replies.

Attack Description

The size of a correctly-formed IPv4 packet including the IP header is 65,535 bytes, including a total payload size of 84 bytes. Many historical computer systems simply could not handle larger packets, and would crash if they received one. This bug was easily exploited in early TCP/IP implementations in a wide range of operating systems including Windows, Mac, Unix, Linux, as well as network devices like printers and routers.

Since sending a ping packet larger than 65,535 bytes violates the Internet Protocol, attackers would generally send malformed packets in fragments. When the target system attempts to

reassemble the fragments, and ends up with an oversized packet, memory overflow could occur and lead to various system problems including crash.

Ping of Death attacks were particularly effective because the attacker's identity could be easily spoofed. Moreover, a Ping of Death attacker would need no detailed knowledge of the machine he/she was attacking, except for its IP address.

It is worthy of note that this vulnerability, though best recognized for its exploitation by PoD attacks, can actually be exploited by anything that sends an IP datagram - ICMP echo, TCP, UDP and IPX.

Methods of Mitigation

To avoid Ping of Death attacks, and its variants, many sites block ICMP ping messages altogether at their firewalls. However, this approach is not viable in the long term. Firstly, invalid packet attacks can be directed at any listening port—like FTP ports—and you may not want to block all of these, for operational reasons. Moreover, by blocking ping messages, you prevent legitimate ping use – and there are still utilities that rely on ping for checking that connections are live, for example.

SYN flood

TCP SYN flood (a.k.a. SYN flood) is a type of Distributed Denial of Service (DDoS) attack that exploits part of the normal TCP three-way handshake to consume resources on the targeted server and render it unresponsive. Essentially, with SYN flood DDoS, the offender sends TCP connection requests faster than the targeted machine can process them, causing network saturation.

Attack Description

When a client and server establish a normal TCP "three-way handshake," the exchange looks like this:

1. Client requests connection by sending SYN (synchronize) message to the server.

2. Server acknowledges by sending SYN-ACK (synchronize-acknowledge) message back to the client.
3. Client responds with an ACK (acknowledge) message, and the connection is established.

In a SYN flood attack, the attacker sends repeated SYN packets to every port on the targeted server, often using a fake IP address. The server, unaware of the attack, receives multiple, apparently legitimate requests to establish communication. It responds to each attempt with a SYN-ACK packet from each open port.

The malicious client either does not send the expected ACK, or—if the IP address is spoofed—never receives the SYN-ACK in the first place. Either way, the server under attack will wait for acknowledgement of its SYN-ACK packet for some time. During this time, the server cannot close down the connection by sending an RST packet, and the connection stays open. Before the connection can time out, another SYN packet will arrive. This leaves an increasingly large number of connections half-open – and indeed SYN Food attacks are also referred to as "half-open" attacks. Eventually, as the server's connection overflow tables fill, service to legitimate clients will be denied, and the server may even malfunction or crash.

Methods of Mitigation

While modern operating systems are better equipped to manage resources, which makes it more difficult to overflow connection tables, servers are still vulnerable to SYN flood attacks. There are a number of common techniques to mitigate SYN flood attacks, including:

Micro blocks—administrators can allocate a micro-record (as few as 16 bytes) in the server memory for each incoming SYN request instead of a complete connection object.

SYN cookies—using cryptographic hashing, the server sends its SYN-ACK response with a sequence number (seqno) that is constructed from the client IP address, port number, and possibly other unique identifying information. When the client responds, this hash is included in the ACK packet. The server verifies the ACK, and only then allocates memory for the connection.

RST cookies—for the first request from a given client, the server intentionally sends an invalid SYN-ACK. This should result in the client generating an RST packet, which tells the server something is wrong. If this is received, the server knows the request is legitimate, logs the client, and accepts subsequent incoming connections from it.

Stack tweaking—administrators can tweak TCP stacks to mitigate the effect of SYN floods. This can either involve reducing the timeout until a stack frees memory allocated to a connection, or selectively dropping incoming connections.

Obviously, all of the above-mentioned methods rely on the target network's ability to handle large-scale volumetric DDoS attacks, with traffic volumes measured in tens of Gigabits (and even hundreds of Gigabits) per second.

Distributed Denial of Service

DDoS *is short for* Distributed Denial of Service. DDoS is a type of DOS attack where multiple compromised systems, which are often infected with a Trojan, are used to target a single system causing a Denial of Service (DoS) attack. Victims of a DDoS attack consist of both the end targeted system and all systems maliciously used and controlled by the hacker in the distributed attack.

How DDoS Attacks Work

In a DDoS attack, the incoming traffic flooding the victim originates from many different sources – potentially hundreds of thousands or more. This effectively makes it impossible to stop the attack simply by blocking a single IP address; plus, it is very difficult to distinguish legitimate user traffic from attack traffic when spread across so many points of origin.

DoS vs DDoS

The differences between DoS and DDoS are substantive and worth noting. In a DoS attack, a perpetrator uses a single Internet connection to either exploit a software vulnerability or flood a target with fake requests—usually in an attempt to exhaust server resources (e.g., RAM and CPU). On the other hand, distributed denial of service (DDoS) attacks are launched from multiple

connected devices that are distributed across the Internet. These multi-person, multi-device barrages are generally harder to deflect, mostly due to the sheer volume of devices involved. Unlike single-source DoS attacks, DDoS assaults tend to target the network infrastructure in an attempt to saturate it with huge volumes of traffic.

DDoS attacks also differ in the manner of their execution. Broadly speaking, DoS attacks are launched using homebrewed scripts or DoS tools (e.g., Low Orbit Ion Canon), while DDoS attacks are launched from botnets—large clusters of connected devices (e.g., cellphones, PCs or routers) infected with malware that allows remote control by an attacker.

Advanced Persistent Threats

Today we are seeing targeted cyber-attacks on organizations grow progressively more sophisticated, more serious, and more extensive. In the mid-2000s, the “black hat” community evolved from adolescent hackers bent on mayhem to organized crime networks, fueling highly profitable identity theft schemes with massive loads of personal data harvested from corporate and government networks. More recently, changes in IT infrastructure and usage models, including mobility, cloud computing, and virtualization have dissolved traditional enterprise security perimeters, creating a "target-rich" environment for hackers. But perhaps the most significant new element in the threat landscape is the emergence of highly targeted, long-term, international espionage and sabotage campaigns by covert state actors. These long-term, state-sponsored campaigns are sometimes known as Advanced Persistent Threats (APTs). The term has become a buzzword used and misused by the media, and by some technology vendors. While APTs do represent a real danger in today’s world, it is important to understand how they figure within a larger context. Only by separating reality from hype and seeing how APTs relate to the broader field of targeted attack methods and techniques will organizations be able to safeguard their information and operations in the coming decade.

What is an APT?

An APT is a type of targeted attack. Targeted attacks use a wide variety of techniques, including drive-by downloads, Microsoft SQL injection, malware, spyware, phishing, and spam, to name just a few. APTs can and often do use many of these same techniques. An APT is always a targeted attack, but a targeted attack is not necessarily an APT.

APTs are different from other targeted attacks in the following ways:

• Customized attacks—In addition to more common attack methods, APTs often use highly customized tools and intrusion techniques, developed specifically for the campaign. These tools include zero-day vulnerability exploits, viruses, worms, and rootkits. In addition, APTs often launch multiple threats or “kill chains” simultaneously to breach their targets and ensure ongoing access to targeted systems, sometimes including a "sacrificial" threat to trick the target into thinking the attack has been successfully repelled.

• Low and slow—APT attacks occur over long periods of time during which the attackers move slowly and quietly to avoid detection. In contrast to the “smash and grab” tactics of many targeted attacks launched by more typical cybercriminals, the goal of the APT is to stay undetected by moving “low and slow” with continuous monitoring and interaction until the attackers achieve their defined objectives.

• Higher aspirations—Unlike the fast-money schemes typical of more common targeted attacks, APTs are designed to satisfy the requirements of international espionage and/or sabotage, usually involving covert state actors. The objective of an APT may include military, political, or economic intelligence gathering, confidential data or trade secret threat, disruption of operations, or even destruction of equipment. The groups behind APTs are well funded and staffed; they may operate with the support of military or state intelligence.

• Specific targets—While nearly any large organization possessing intellectual property or valuable customer information is susceptible to targeted attacks, APTs are aimed at a much smaller range of targets. Widely reported APT attacks have been launched at government agencies and facilities, defense contractors, and manufacturers of products that are highly competitive on global

markets. In addition, APTs may attack vendor or partner organizations that do business with their primary targets. But government-related organizations and manufacturers are not the only targets. Ordinary companies with valuable technology or intellectual property are now being targeted by nation-states. With the globalization of world economies, national security and economic security have converged. Moreover, organizations that maintain and operate vital national infrastructure are also likely targets.

How relevant are APTs?

It should now be evident that although not every organization is a likely target of an APT, they are a real and serious threat to some organizations. Additionally, any organization can benefit from better understanding of APTs, because APT techniques are likely to be adopted over time by mainstream hackers and cybercriminals. Finally, since anyone could be the object of a targeted attack—and APTs are examples of highly advanced, long-term, and large-scale targeted attacks—if you have a better understanding of APTs, you can better defend your organization against targeted threats of any kind.

How do APT attacks work?

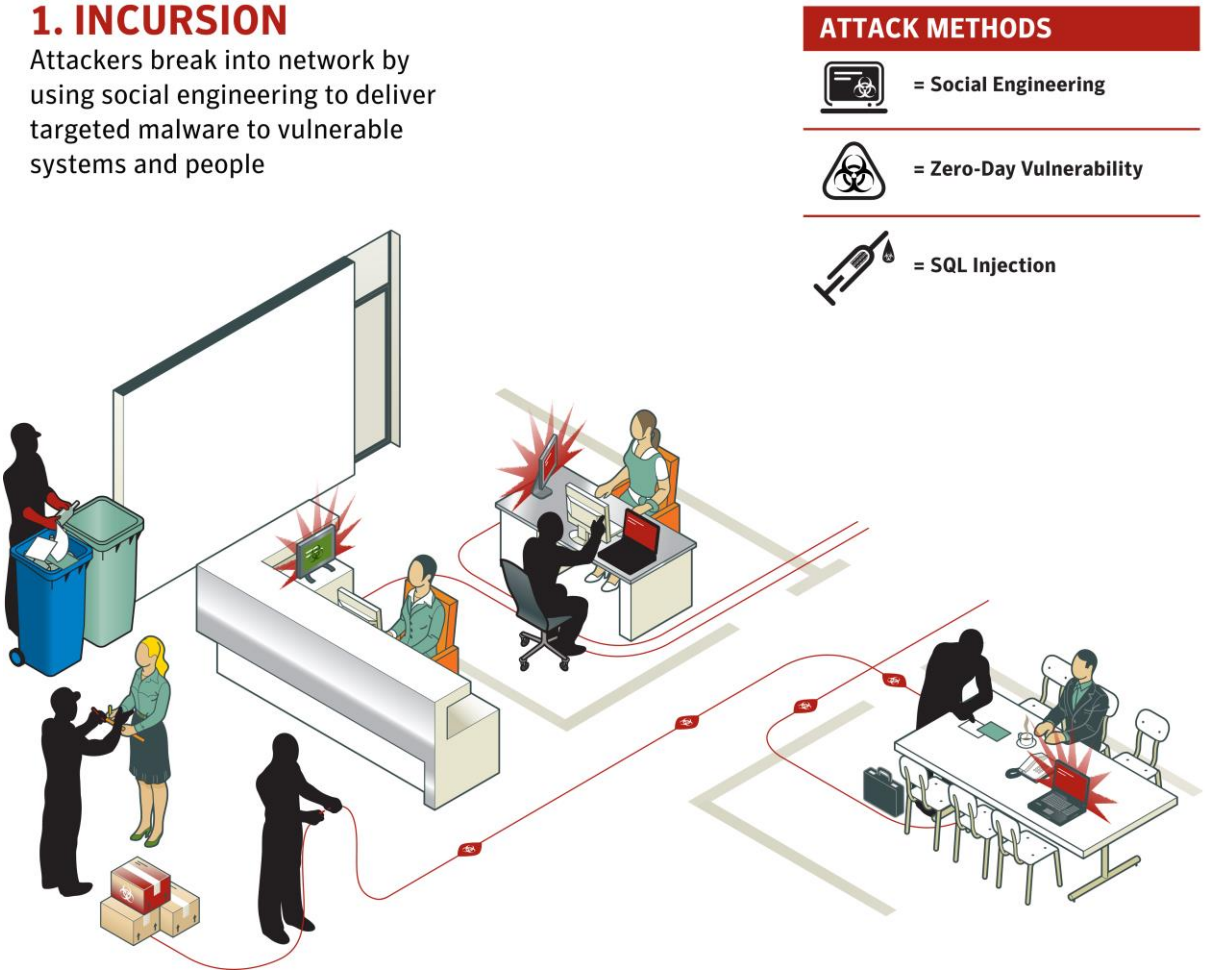
APT attacks are carefully planned and meticulously executed. They typically break down into four phases: incursion, discovery, capture, and exfiltration. In each phase a variety of techniques may be used, as described below.

Phase 1: Incursion

In targeted attacks, hackers typically break into the organization's network using social engineering, zero-day vulnerabilities, SQL injection, targeted malware, or other methods. These methods are also used in APTs, often in concert. The main difference is that while common targeted attacks use short-term, “smash and grab” methods, APT incursions are designed to establish a beach head from which to launch covert operations over an extended period of time. Other characteristics of APT incursions include the following:

Reconnaissance—APT attacks often employ large numbers of researchers who may spend months studying their targets and making themselves familiar with target systems, processes, and people, including partners and vendors. Information may be gathered both online and using conventional surveillance methods. In the case of the Stuxnet attack on organizations believed to be operating Iranian nuclear facilities, the attack team possessed expertise in the design of the programmable logic controllers (PLCs) used for uranium enrichment that were targeted in the attack.

Social engineering—Incursion is often accomplished through the use of social engineering techniques, such as inducing unsuspecting employees to click on links or open attachments that appear to come from trusted partners or colleagues. Unlike the typical phishing attack, such techniques are often fed by in depth research on the target organization. In one case, a small number of human resource employees were targeted using an apparently innocuous attachment, a spreadsheet on hiring needs that appeared to come from a job listing website. In the case of Hydraq, targeted users were led to a picture-hosting website where they were infected via a drive-by-download.



Zero-day vulnerabilities—Zero-day vulnerabilities are security loopholes that are unknown to the software developer and may therefore be exploited by attackers before the developer can provide a patch or fix. As a result, the target organization has zero days to prepare; it is caught off-guard. Since it takes significant time and effort to discover zero-day vulnerabilities, only the most sophisticated attacker organizations are likely to take advantage of them. APTs often use one zero-day vulnerability to breach the target, switch to a second and then a third as each point of attack is eventually fixed. This was the case with Hydraq. The Stuxnet attack was exceptional in that four separate zero-day vulnerabilities were exploited simultaneously.

Manual operations—Common or massive attacks employ automation to maximize their reach. "Spray and pray" phishing scams use automated spam to hit thousands of users in hopes that a certain percentage will click on a link or attachment and trigger the incursion. On the other hand, while APTs may deploy spam, more often they target distinct individual systems and the incursion process is tightly focused—not the automated process used in non-APT attacks.

Phase 2: Discovery

Once inside, the attacker maps out the organization's systems and automatically scans for confidential data or, in the case of some APTs, operational instructions and functionality. Discovery may include unprotected data and networks as well as software and hardware vulnerabilities, exposed credentials, and pathways to additional resources or access points. Here again, where most targeted attacks are opportunistic, APT attacks are more methodical and go to extraordinary lengths to avoid detection.

Multiple vectors—As with incursion, APTs tend to use multiple discovery techniques in combination. Once malware is present on host systems, additional tools can be downloaded as needed for the purpose of exploring software, hardware, and network vulnerabilities.

Run silent, run deep—Since the goal of the APT is to remain inside the organization and harvest information over the long-term, discovery processes are designed to avoid detection at all cost. Hydraq (also known as the Aurora or Google attacks) used a number of obfuscation techniques to keep itself hidden inside victim organizations. Specifically, it used spaghetti code, a technique used to make analysis and detection of the malware more difficult.

Research and analysis—Discovery efforts are accompanied by research and analysis on found systems and data, including network topology, user IDs, passwords, and so on. When an APT is detected, the first question asked is: How long has it been there? Not so with the typical targeted attack; if account numbers have been stolen it is not difficult to date the breach and assess the damage. With APTs, however, it may be next to impossible to determine just when the attack took place. Victims may need to comb through log files or even dispose of equipment because the

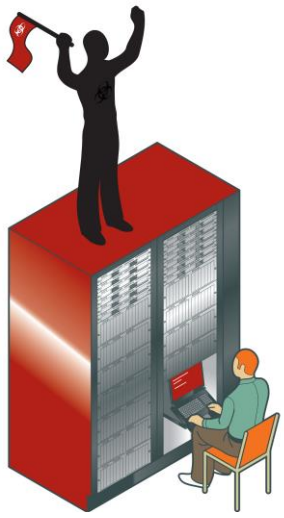
incursion and discovery phases have been so well hidden. In some cases, the APT kill-chain may be quite easy to find. But appearances can be deceptive. The obvious kill-chain may be intentionally launched to distract the victim while the perpetrators proceed undetected to their actual objectives.

Phase 3: Capture

In the capture phase, exposed data stored on unprotected systems is immediately accessed. In addition, rootkits may be surreptitiously installed on targeted systems and network access points to capture data and instructions as they flow through the organization. In the case of Duqu, which seems to be the precursor to a future, Stuxnet-like attack, its sole purpose was to gather intelligence, which could be used to give attackers the insight they need to mount future attacks.

While Duqu was not widespread, it is highly targeted, and its targets include suppliers to industrial facilities.

Long-term occupancy—The APT is designed to capture information over an extended period. For example, a large-scale cyber spying operation called GhostNet, discovered in March 2009, was able to infiltrate computer systems in 103 countries, including embassies, foreign ministries, and other government offices, and the Dalai Lama's Tibetan exile centers in India, London, and New York City. According to a report by the Information Warfare Monitor, GhostNet began capturing data on May 22, 2007, and continued at least through March 12, 2009. On average, the amount of time that a host was actively infected by an APT was 145 days, with the longest infection span being 660 days.

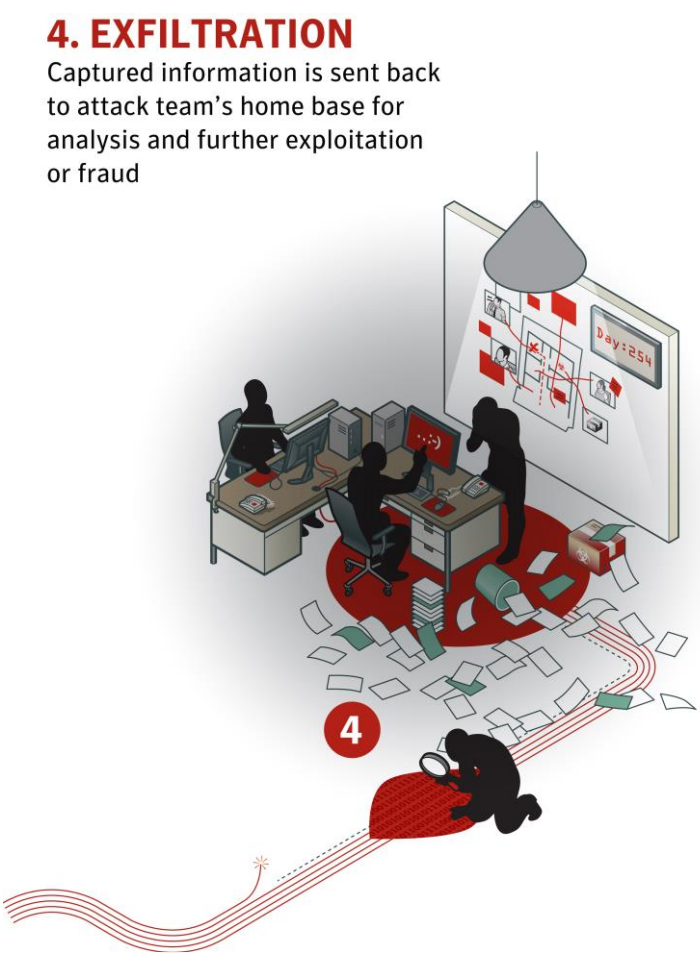


Control—In some cases, APTs entail the remote ignition or shutdown of automated software and hardware systems. As more and more physical devices are controlled by embedded microprocessors, the potential for mayhem is high. In fact, Stuxnet went well beyond stealing information. Its purpose was to reprogram industrial control systems—computer programs used to manage industrial environments such as power plants, oil refineries, and gas pipelines. Specifically, its goal was to manipulate the physical equipment attached to specific industrial control systems so the equipment acted in a manner programmed by the attacker, contrary to its intended purpose. Command-and-control servers may covertly seize control of target systems and even destroy them depending on the APT game plan.

Phase 4: Exfiltration

Once the intruders have seized control of target systems, they may proceed with the theft of intellectual property or other confidential data.

Data transmission—Following command-and-control signals, harvested data may be sent back to the attack team home base either in the clear (by Web mail, for example) or wrapped in encrypted packets or zipped files with password protection. Hydraq used a number of novel techniques for sending the stolen information back to home base. One of these was the use of Port 443 as a primary channel for upload of stolen data. It also established connections that resembled an SSL key exchange dialogue, but did not result in a fully negotiated SSL channel. Lastly, it used private ciphers to encrypt content as it left the victim organizations.



Ongoing analysis—Whereas stolen credit card numbers from a targeted attack are quickly packaged for sale, information captured by APTs is often studied at length for clues to strategic opportunities. Such data may be subject to manual analysis by field experts to extract trade secrets, anticipate competitive moves, and plan counter maneuvers.

What to do?

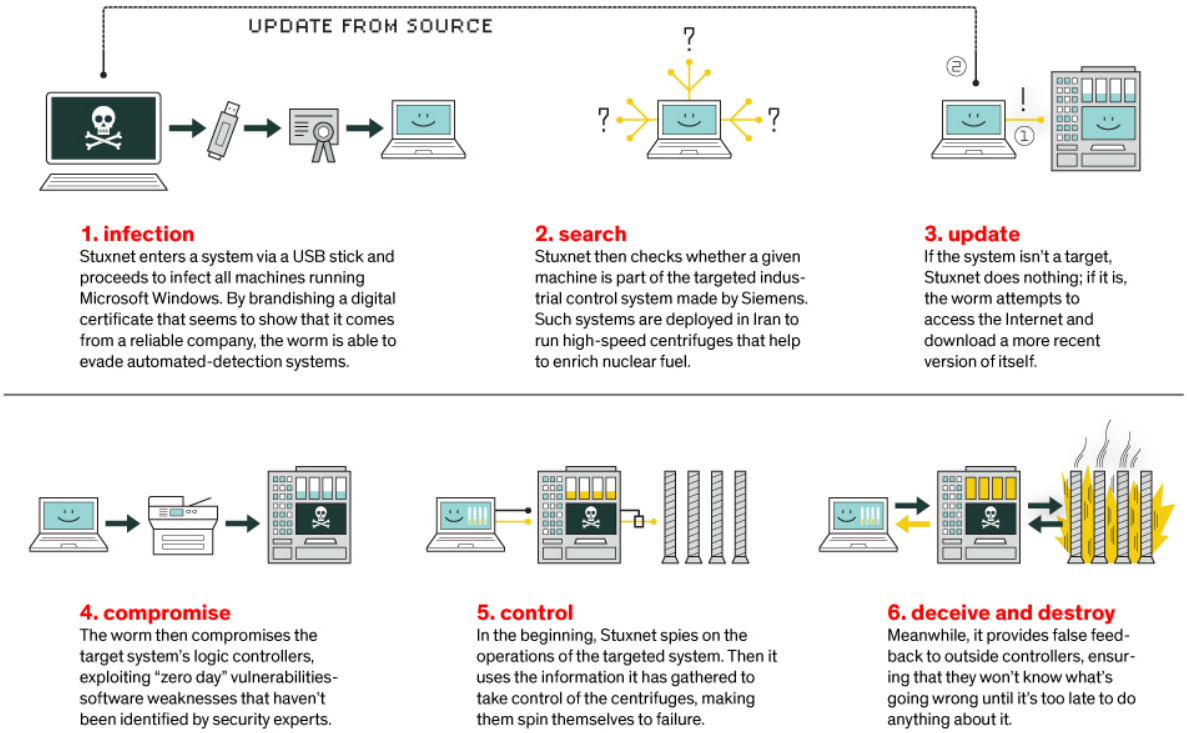
The hype surrounding APTs masks an underlying reality—these threats are, in fact, a special case within the much broader category of attacks targeted at specific organizations of all kinds. As APTs continue to appear on the threat landscape—and there is no reason to think that they will not—we expect to see the same techniques deployed by other cybercriminals. Moreover, the fact that APTs are often aimed at stealing intellectual property suggests new roles for cybercriminals as information brokers in industrial espionage schemes.

The best way to prepare for an APT is to ensure you are well defended against targeted attacks in general. In fact, while the odds of an APT affecting your organization may be relatively low, the chances that you may be the victim of a targeted attack are, unfortunately, quite high.

Stuxnet

At some point during the first half of 2009, a 500kb sized copy of the stuxnet malware resided in an inconspicuous USB drive. This drive incidentally came into the possession of an individual with basic user privileges at Natanz nuclear facility, Iran. Soon, the USB drive was inserted into a machine within the local network, where the software took advantage of a little-known software vulnerability involving .LNK extensions to copy itself to the stored data files. It was then uploaded to the machine, perhaps under the nose of the individual who possessed the drive, where it was automatically executed and hidden by rootkit components. The designers of stuxnet provided it with the ability to hide with encryption keys, which yield “certificates of authenticity” in order to disguise the payload as software that was created by JMicron or Realtek. As the program distributed itself to machines across the network through file sharing and/or a vulnerability in the print spooler remote code execution service (Kiezer 2015), it also fingerprinted each system to determine if they run Step 7 Programmable Logic Controller (PLC) executables. By doing so, it mapped the entire network for future reference. If stuxnet happened to find itself on a web-facing machine, it would update itself through a peer-to-peer network if it needed to adapt to the conditions of the network map. While the apparatus of stuxnet proved to be sophisticated and

reliable, its payload was ultimately just designed to manipulate the Siemens S7 – 400 PLCs that govern and monitor centrifuges meant for uranium enrichment. It did so by covertly altering the programmable values related to centrifuge rotation, while performing a man-in-the-middle attack to mislead technicians into believing that the PLCs were correctly configured; if the technician tried to shut the centrifuges down, stuxnet would then have disabled their ability to abort. By the end of this particular attack, Natanz suffered the loss of around 1000 centrifuges at the hands of stuxnet, stalling Iran's uranium enrichment schedule by approximately 2 years (Zetter 2014).



Stuxnet impacted the world in a broader sense, as well. Outside the Natanz facility, over 30,000 Iranian machines were reported to be infected by stuxnet in 2010. It stands to reason that the creators of the malware may have disseminated it to increase the likelihood of it infecting machines that could be linked to services provided to local uranium enrichment centers. This unusually large piece of malware also acted out 4 zero-day exploits (attacks against previously unknown vulnerabilities), in a world where malicious bundles of software tend to prey upon 1

zero-day each (Kushner 2015, Langner 2013, Naraine 2010) . Furthermore, the code lingers on the net with open-source availability, where programmers can freely download it and augment it.

Malwares in Other Operating Systems

We all know that Windows is the most malware-ridden platform out there, but why is that? Windows is the most popular desktop operating system, but that isn't the only reason – past decisions made Windows a fertile breeding ground for viruses and other malware.

Windows is a big target because it powers the vast majority of the world's desktop computers and laptops. If you're writing malware and you want to infect average computers users – perhaps you want to install a key logger on their systems and steal their credit card numbers and other financial data – you would target Windows because that's where the most users are.

Historically, Windows was not designed for security. While Linux and Apple's Mac OS X (based on Unix) were built from the ground-up to be multi-user operating systems that allowed users to log in with limited user accounts, the original versions of Windows never were.

DOS was a single-user operating system, and the initial versions of Windows were built on top of DOS. Windows 3.1, 95, 98, and Me may have looked like advanced operating systems at the time, but they were actually running on top of the single-user DOS. DOS didn't have proper user accounts, file permissions, or other security restrictions.

Windows NT – the core of Windows 2000, XP, Vista, 7, and now 8 – is a modern, multi-user operating system that supports all the essential security settings, including the ability to restrict user account permissions. However, Microsoft never really designed consumer versions of Windows for security until Windows XP SP2. Windows XP supported multiple user accounts with limited privileges, but most people just logged into their Windows XP systems as the Administrator user. Much software wouldn't work if you did use a limited user account, anyway. Windows XP shipped without a firewall enabled and network services were exposed directly to the Internet, which made it an easy target for worms. At one point, the SANS Internet Storm Center

estimated an unpatched Windows XP system would be infected within four minutes of connecting it directly to the Internet, due to worms like Blaster.

In addition, Windows XP's autorun feature automatically ran applications on media devices connected to the computer. This allowed Sony to install a rootkit on Windows systems by adding it to their audio CDs, and savvy criminals began leaving infected USB drives lying around near companies they wanted to compromise. If an employee picked up the USB drive and plugged it into a company computer, it would infect the computer. And, because most users logged in as Administrator users, the malware would run with administrative privileges and have complete access to the computer. It's clear that Microsoft never designed the original release of Windows XP to survive on a dangerous Internet, and it showed. The lack of an official app store for desktop application also increases the risk for less-savvy computer users looking for software online. Users that don't know the warning signs and what to avoid are much more vulnerable on the Windows desktop.

Linux malware

Linux malware includes viruses, trojans, worms and other types of malware that affect the Linux operating system. Linux, Unix and other Unix-like computer operating systems are generally regarded as very well-protected against, but not immune to, computer viruses. There has not yet been a single widespread Linux virus or malware infection of the type that is common on Microsoft Windows; this is attributable generally to the malware's lack of root access and fast updates to most Linux vulnerabilities.

Linux vulnerability

Like Unix systems, Linux implements a multi-user environment where users are granted specific privileges and there is some form of access control implemented. To gain control over a Linux system or to cause any serious consequences to the system itself, the malware would have to gain root access to the system. In the past, it has been suggested that Linux had so little malware because its low market share made it a less profitable target. In 2008 the quantity of malware targeting Linux

was noted as increasing. Shane Coursen, a senior technical consultant with Kaspersky Lab, said at the time, "The growth in Linux malware is simply due to its increasing popularity, particularly as a desktop operating system ... The use of an operating system is directly correlated to the interest by the malware writers to develop malware for that OS".

Some Linux users do run Linux-based anti-virus software to scan insecure documents and email which comes from or is going to Windows users. Because they are predominantly used on mail servers which may send mail to computers running other operating systems, Linux virus scanners generally use definitions for, and scan for, all known viruses for all computer platforms.

Viruses and trojan horses

If an infected binary containing one of the viruses were run, the system would be temporarily infected; Linux kernel is memory resident and read-only. Any infection level would depend on which user with what privileges ran the binary. A binary run under the root account would be able to infect the entire system. Privilege escalation vulnerabilities may permit malware running under a limited account to infect the entire system. It is worth noting that this is true for any malicious program that is run without special steps taken to limit its privileges. It is trivial to add a code snippet to any program that a user may download and let this additional code download a modified login server, an open mail relay, or similar program, and make this additional component run any time the user logs in. No special malware writing skills are needed for this. Special skill may be needed for tricking the user to run the (trojan) program in the first place. The use of software repositories significantly reduces any threat of installation of malware, as the software repositories are checked by maintainers, who try to ensure that their repository is malware-free. Subsequently, to ensure safe distribution of the software, checksums are made available. These make it possible to reveal modified versions that may have been introduced by e.g. hijacking of communications using a man-in-the-middle attack or via a redirection attack such as ARP or DNS poisoning. Careful use of these digital signatures provides an additional line of defense, which limits the scope of attacks to include only the original authors, package and

release maintainers and possibly others with suitable administrative access, depending on how the keys and checksums are handled.

Worms and targeted attacks

The classical threat to Unix-like systems are vulnerabilities in network daemons, such as SSH and web servers. These can be used by worms or for attacks against specific targets. As servers are patched quite quickly when a vulnerability is found, there have been only a few widespread worms of this kind. As specific targets can be attacked through a vulnerability that is not publicly known there is no guarantee that a certain installation is secure. Also servers without such vulnerabilities can be successfully attacked through weak passwords.

Web scripts

Linux servers may also be used by malware without any attack against the system itself, where e.g. web content and scripts are insufficiently restricted or checked and used by malware to attack visitors. Some attacks use complicated malware to attack Linux servers, but when most get full root access then hackers are able to attack by modifying anything like replacing binaries or injecting modules. This may allow the redirection of users to different content on the web. Typically, a CGI script meant for leaving comments, could, by mistake, allow inclusion of code exploiting vulnerabilities in the web browser.

Buffer overruns

Older Linux distributions were relatively sensitive to buffer overrun attacks: if the program did not care about the size of the buffer itself, the kernel provided only limited protection, allowing an attacker to execute arbitrary code under the rights of the vulnerable application under attack. Programs that gain root access even when launched by a non-root user (via the setuid bit) were particularly attractive to attack. However, as of 2009 most of the kernels include address space layout randomization (ASLR), enhanced memory protection and other extensions making such attacks much more difficult to arrange.

Cross-platform viruses

An area of concern identified in 2007 is that of crossplatform viruses, driven by the popularity of cross-platform applications. This was brought to the forefront of malware awareness by the distribution of an OpenOffice.org virus called Badbunny.

Linux.Lady TROJAN

The Linux.Lady malware was discovered by Russian antivirus software vendor Dr Web and is, intriguingly, written using Google's Go programming language, largely based on open source Go libraries hosted on GitHub.

The malware uses a more compact trojan called Linux.Downloader.196 to download the main payload after infection. Linux.Lady, once installed and running, sends basic information about the cracked system to the command-and-control (C&C) server. The next step in the infection process is a configuration file sent from the C&C server to start the crypto-currency mining process for the benefit of the malware's controllers. Linux.Lady is also self-propagating. "This malware possesses the ability to collect information about an infected computer and transfer it to the C&C server, download and launch a crypto-currency mining utility, and attack other computers on the network to install its own copy on them," said the Dr Web advisory. Once launched, the trojan checks the system for keys and terminates itself if they are missing:

- Version - display the trojan's version and terminate the session
- Install - install the trojan
- D - launch main payload of the trojan.

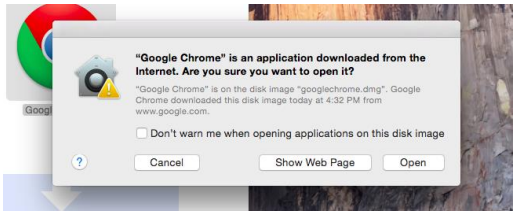
MAC OS

Apple introduced malware detection to the Mac OS with Snow Leopard (Mac OS 10.6). This system consists of the quarantine of any app downloaded from the Internet, the use of Code Signing

certificates to verify that an app is coming from a legit source, and regular security updates that include databases of known malware targeting the Mac OS.

XProtect

The built-in anti-malware protection on Mac OS X is known as "XProtect," which is technically a feature built into "File Quarantine." This feature was added back in 2009 with Mac OS X 10.6 Snow Leopard. When you open, an application downloaded from the Internet using a "File Quarantine-aware" application like Safari, Chrome, Mail, or iChat, you'll see a warning message informing you the application was downloaded from the web along with the specific website it was downloaded from and when. It's a bit like the "This application was downloaded from the Internet!" warning dialogs you'll see after downloading and trying to run an application on Windows.



Back in 2009, Apple made File Quarantine also check downloaded application files against a list stored in the System/Library/Core Services/CoreTypes.bundle/Contents/Resources/XProtect.plist file on your Mac. You can even open this file and see the list of malicious applications Mac OS X is checking for when you open downloaded application files.

Mobile Malware

Mobile malware is malicious software that targets mobile phones or wireless-enabled Personal digital assistants (PDA), by causing the collapse of the system and loss or leakage of confidential information. As wireless phones and PDA networks have become more and more common and have grown in complexity, it has become increasingly difficult to ensure their safety and security against electronic attacks in the form of viruses or other malware.

History

Cell phone malware were initially demonstrated by Brazilian software engineer Marcos Velasco. He created a virus that could be used by anyone in order to educate the public of the threat. The first known mobile virus, "Timofonica", originated in Spain and was identified by antivirus labs in Russia and Finland in June 2000. "Timofonica" sent SMS messages to GSM mobile phones that read (in Spanish) "Information for you: Telefónica is fooling you." These messages were sent through the Internet SMS gate of the MoviStar mobile operator. In June 2004, it was discovered that a company called Ojam had engineered an anti-piracy Trojan virus in older versions of its mobile phone game, Mosquito. This virus sent SMS text messages to the company without the user's knowledge. Although this malware was removed from the game's more recent versions, it still exists in older, unlicensed versions, and these may still be distributed on file-sharing networks and free software download web sites. In July 2004, computer hobbyists released a proof-of-concept mobile virus Cabir, that replicates and spreads itself on Bluetooth wireless networks and infects mobile phones running the Symbian OS. In March 2005, it was reported that a computer worm called Commwarrior-A had been infecting Symbian series 60 mobile phones. This specific worm replicated itself through the phone's Multimedia Messaging Service (MMS), sending copies of itself to other phone owners listed in the phone user's address book. Although the worm is not considered harmful, experts agree that it heralded a new age of electronic attacks on mobile phones. In August 2010, Kaspersky Lab reported a trojan designated Trojan-SMS.AndroidOS.FakePlayer.a. This was the first malicious program classified as a Trojan SMS that affects smartphones running on Google's Android operating system, and which had already infected a number of mobile devices, sending SMS messages to premium rate numbers without the owner's knowledge or consent, and accumulating huge bills.

Notable mobile malicious programs

Cabir: This malware infects mobile phones running on Symbian OS and was first identified in June 2004. When a phone is infected, the message 'Caribe' is displayed on the phone's screen and is displayed every time the phone is turned on. The worm then attempts to spread to other phones in the area using wireless Bluetooth signals, although the recipient has to confirm this manually.

Skulls: A trojan horse piece of code that targets mainly Symbian OS. Once downloaded, the virus replaces all phone desktop icons with images of a skull. It also renders all phone applications useless. This malware also tends to mass text messages containing malicious links to all contacts accessible through the device in order to spread the damage. This mass texting can also give rise to high expenses.

Commwarrior: This malware was identified in 2005. It was the first worm to use MMS messages in order to spread to other devices. It can spread through Bluetooth as well. It infects devices running under OS Symbian Series 60. The executable worm file, once launched, hunts for accessible bluetooth devices and sends the infected files under a random name to various devices.

Gingermaster: A trojan developed for an Android platform that propagates by installing applications that incorporate a hidden malware for installation in the background. It exploits the frailty in the version Gingerbread (2.3) of the operating system to use superuser permissions by privileged escalation. Then it creates a service that steals information from infected terminals (user ID, number SIM, phone number, IMEI, IMSI, screen resolution and local time) by sending it to a remote server through petitions HTTP.

DroidKungFu: A trojan content in Android applications, which when executed, obtains root privileges and installs the file com.google. ssearch.apk, which contains a back door that allows files to be removed, open home pages to be supplied, and 'open web and download and install' application packages. This virus collects and sends to a remote server all available data on the terminal.

Ikee: The first worm known for iOS platforms. It only works on terminals that were previously made a process of jailbreak, and spreads by trying to access other devices using the SSH protocol, first through the subnet that is connected to the device. Then, it repeats the process generating a random range and finally uses some preset ranges corresponding to the IP address of certain telephone companies. Once the computer is infected, the wallpaper is replaced by a photograph of the singer Rick Astley, a reference to the Rickroll phenomenon.

Stagefright: This past summer in the days leading up to the annual Black Hat security conference in Las Vegas, a number of vulnerabilities were found in the Android operating systems. This collection of bugs was referred to as "Stagefright" in reference to the stagefright libraries (underlying code in the OS that is shared by many applications) contained in the Android OS. These vulnerabilities are particularly nasty due to the fact that they allow an attacker to remotely execute code on someone's phone by sending a specially crafted MMS message. Typically, a cybercriminal

will try to trick the user into clicking on a malicious Web link or installing an infected application. None of those steps are necessary with Stagefright. If someone knows the intended target's phone number, that's all they need to launch an attack. Scarier still, due to the nature of the bugs, it would be possible for an attacker to hack a phone, implant a remote access tool, and cover any trace that the attack had occurred; all while the phone was charging overnight on the victim's nightstand.

Protection Against Malware

Sandboxing

A sandbox is a safe isolated environment that replicates an end user's operating environment where one can run binaries, observe the environment and rate it based on activity rather than attributes. This is an effective way and looking at a behavior of a file before passing it on to the actual environment provides a higher level of protection. It is often used to execute untested or untrusted programs or code, possibly from unverified or untrusted third parties, suppliers, users or websites, without risking harm to the host machine or operating system.

A sandbox typically provides a tightly controlled set of resources for guest programs to run in, such as scratch space on disk and memory. Network access, the ability to inspect the host system or read from input devices are usually disallowed or heavily restricted. In the sense of providing a highly-controlled environment, sandboxes may be seen as a specific example of virtualization. Sandboxing is frequently used to test unverified programs that may contain a virus or other malicious code, without allowing the software to harm the host device.

The problem with sandboxing is that malware today can lie dormant for weeks or they can activate on a particular series of events that is never triggered inside the sandbox. This means that the sandbox environment will perceive the file to be safe, when in fact it is just waiting to be transferred to the real environment before attacking.

Antivirus Software

Antivirus or anti-virus software (often abbreviated as AV), sometimes known as anti-malware software, is computer software used to prevent, detect and remove malicious software. Antivirus software was originally developed to detect and remove computer viruses, hence the name. However, with the proliferation of other kinds of malware, antivirus software started to provide protection from other computer threats. In particular, modern antivirus software can protect from: malicious browser helper objects (BHOs), browser hijackers, ransomware, keyloggers, backdoors, rootkits, trojan horses, worms, malicious LSPs, dialers, fraudtools, adware and spyware. Some products also include protection from other computer threats, such as infected

and malicious URLs, spam, scam and phishing attacks, online identity (privacy), online banking attacks, social engineering techniques, advanced persistent threat (APT) and botnet DDoS attacks.

Although the roots of the computer virus date back as early as 1949, when the Hungarian scientist John von Neumann published the “Theory of self-reproducing automata”, the first known computer virus appeared in 1971 and was dubbed the "Creeper virus". This computer virus infected Digital Equipment Corporation's (DEC) PDP-10 mainframe computers running the TENEX operating system. The Creeper virus was eventually deleted by a program created by Ray Tomlinson and known as “The Reaper”. Some people consider “The Reaper” the first antivirus software ever written – it may be the case, but it is important to note that the Reaper was actually a virus itself specifically designed to remove the Creeper virus. The Creeper virus was followed by several other viruses. The first known that appeared “in the wild” was "Elk Cloner", in 1981, which infected Apple II computers.

The first IBM PC compatible “in the wild” computer virus, and one of the first real widespread infections, was "Brain" in 1986. From then, the number of viruses has grown exponentially. Most of the computer viruses written in the early and mid-1980s were limited to selfreproduction and had no specific damage routine built into the code. That changed when more and more programmers became acquainted with computer virus programming and created viruses that manipulated or even destroyed data on infected computers. Before internet connectivity was widespread, computer viruses were typically spread by infected floppy disks. Antivirus software came into use, but was updated relatively infrequently. During this time, virus checkers essentially had to check executable files and the boot sectors of floppy disks and hard disks. However, as internet usage became common, viruses began to spread online.

Over the years, it has become necessary for antivirus software to use several different strategies (e.g. specific email and network protection or low level modules) and detection algorithms, as well as to check an increasing variety of files, rather than just executables, for several reasons:

- Powerful macros used in word processor applications, such as Microsoft Word, presented a risk. Virus writers could use the macros to write viruses embedded within

documents. This meant that computers could now also be at risk from infection by opening documents with hidden attached macros.

- The possibility of embedding executable objects inside otherwise non-executable file formats can make opening those files a risk.
- Later email programs, in particular Microsoft's Outlook Express and Outlook, were vulnerable to viruses embedded in the email body itself. A user's computer could be infected by just opening or previewing a message.

Detection Techniques

Specific Detection

Specific detection is what a lot of people think of when they think of anti-virus scanners. It looks for known malware by a specific set of characteristics. Each malware uses its own code to do its thing. To detect malware specifically, the scanner looks for that signature in a fairly particular place. This technique can be fast, because it can exclude clean files pretty quickly if a researcher does things right, but it's also fairly easy to evade this sort of detection. Change the code, move it somewhere else, encrypt it, or hide it in some other way, and then the threat doesn't get detected anymore.

As malware is getting much more crafty and prevalent, researchers must get more creative to quickly identify known-bad files, lest they be overwhelmed by the ton of samples that arrive every day. So, the next step is to group samples by what's called "families," which means they're related by a common code-base.

Generic Detection

Generic detection looks for malware that are variants of known families, which are often created by a common group of programmers. Within a generic detection, there is usually common functionality and occasionally common signatures, and detection is meant to catch both known malware samples as well as new samples based on that known code-base.

Since malware is all about the money these days, the best way for malware authors to get the biggest bang for buck is to reuse their code-base. They may add or change minor functionality,

or they may just move things around to evade specific detection. There are a lot of different, common malware that have freely available, open-source software or malware-creation kits that allow for quick and easy use.

So, it makes sense for anti-virus scanners to look for common properties of those popular malware families or known malicious behavior if they want to be have any hope of keeping up. These generic detections can be fairly broad or fairly specific. For example, it could scan for known exploit code that could be added to known malware or brand-new creations. Or it could look for specific packers that are used by only one malware family.

Heuristic Detection

Heuristic detection is scanning for previously unknown viruses by looking for known suspicious behavior or file structures. This is where scanning gets more broad and speculative. Basically, heuristic detection applies the “smell test” to files. Is there anything about this file that looks hinky? Are its structures odd in a way that would imply that it’s trying to hide something? Does it behave in a way that benign files generally don’t?

Heuristic scanners usually add up a number of things, possibly positive and definitely negative, in order to give the program a rating. If the rating leans towards identifying a lot of bad behavior, it’s likely the file is malicious. Sometimes a scanner determines the rating by viewing the file statically (on disk) or dynamically (in action, in a virtual environment or a sandbox). Some scanners will allow you to dial up or down your preferred level of paranoia.

Micro – Virtualization technology - Bromium

A Web browser like Internet Explorer, Firefox, or Chrome — it's isolated into its own virtual machine called a "Micro-VM". A Micro-VM puts the application on a "need-to-know" basis, and only provisions out exactly what it needs in order to function. For example, it doesn't have access to every library on the system; only the ones that it needs to run. Applications may have multiple processes running within them, such as multiple tabs in a Web browser. In this case, a browser tab as well as any plugins inside them would be given their own Micro-VM. There are no "child" virtual

machine processes, only parallel Micro-VM processes, all running within the microvisor's "ring of trust".

Now here is where things get interesting. When the application or the process within the application is closed, that Micro-VM also dies. Any malware that may have entered the system via that process is destroyed along with it. Also, If you are visiting a website and get hit with a redirect/cross-site scripting or a phishing attempt, it employs what the company(Bromium) refers to as Live Attack Visualization and Analysis (LAVA), which uses the behavioral signature of the attack to determine that it needs to shut down the virtual machine and notify the user before the compromise actually occurs. This includes sophisticated malware attacks including those that utilize polymorphism as well as rootkits and boot-kits.

Honeypot

In computer terminology, a honeypot is a computer security mechanism set to detect, deflect, or, in some manner, counteract attempts at unauthorized use of information systems. Generally, a honeypot consists of data (for example, in a network site) that appears to be a legitimate part of the site but is actually isolated and monitored, and that seems to contain information or a resource of value to attackers, which are then blocked. This is similar to the police baiting a criminal and then conducting undercover surveillance, and finally punishing the criminal.

Types

Honeypots can be classified based on their deployment (use/action) and based on their level of involvement. Based on deployment, honeypots may be classified as

1. Production Honeypots
2. Research Honeypots

Production honeypots are easy to use, capture only limited information, and are used primarily by companies or corporations. Production honeypots are placed inside the production network with other production servers by an organization to improve their overall state of security. Normally, production honeypots are low-interaction honeypots, which are easier to deploy. They give less

information about the attacks or attackers than research honeypots. Research honeypots are run to gather information about the motives and tactics of the Black hat community targeting different networks. These honeypots do not add direct value to a specific organization; instead, they are used to research the threats that organizations face and to learn how to better protect against those threats. Research honeypots are complex to deploy and maintain, capture extensive information, and are used primarily by research, military, or government organizations.

Honeynets

Two or more honeypots on a network form a honeynet. Typically, a honeynet is used for monitoring a larger and/or more diverse network in which one honeypot may not be sufficient. Honeynets and honeypots are usually implemented as parts of larger network intrusion detection systems. A honeyfarm is a centralized collection of honeypots and analysis tools.

The concept of the honeynet first began in 1999 when Lance Spitzner, founder of the Honeynet Project, published the paper "To Build a Honeypot":

"A honeynet is a network of high interaction honeypots that simulates a production network and configured such that all activity is monitored, recorded and in a degree, discreetly regulated."

Malware Analysis

Malware Analysis is the study or process of determining the functionality and potential impact of a given malware sample such as a virus, trojan horse, rootkit, or backdoor. Malware or malicious software is any computer software intended to harm the host operating system or to steal sensitive data from users, organizations or companies. Malware may include viruses, worms, trojan horses, and spyware that gathers user information without permission.

Malware analysis use cases

There are three typical use cases that drive the need for malware analysis:

1. **Computer security incident management**: If an organization discovers or suspects that some malware may have gotten into its systems, a response team may wish to perform malware analysis on any potential samples that are discovered during the investigation

process to determine if they are malware and, if so, what impact that malware might have on the systems within the target organizations' environment.

2. Malware research: Academic or industry malware researchers may perform malware analysis simply to understand how malware behaves and the latest techniques used in its construction.

3. Indicator of compromise extraction: Vendors of software products and solutions may perform bulk malware analysis in order to determine potential new indicators of compromise; this information may then feed the security product or solution to help organizations better defend themselves against attack by malware.

Malware analysis types

The method by which malware analysis is performed typically falls under one of two types:

1. *Static Malware Analysis:* Static or Code Analysis is usually performed by dissecting the different resources of the binary file without executing it and studying each component. The binary file can also be disassembled (or reverse engineered) using a disassembler such as IDA. The machine code can sometimes be translated into assembly code which can be read and understood by humans: the malware analyst can then make sense of the assembly instructions and have an image of what the program is supposed to perform. Some modern malware is authored using evasive techniques to defeat this type of analysis, for example by embedding syntactic code errors that will confuse disassemblers but that will still function during actual execution.[3]

2. *Dynamic Malware Analysis:* Dynamic or Behavioral analysis is performed by observing the behavior of the malware while it is actually running on a host system. This form of analysis is often performed in a sandbox environment to prevent the malware from actually infecting production systems; many such sandboxes are virtual systems that can easily be rolled back to a clean state after the analysis is complete. The malware may also be debugged while running using a debugger such as GDB or WinDbg to watch the behavior and effects on the host system of the malware step by step while its instructions are being

processed. Modern malware can exhibit a wide variety of evasive techniques designed to defeat dynamic analysis including testing for virtual environments or active debuggers, delaying execution of malicious payloads, or requiring some form of interactive user input.

Four stages of malware analysis

Examining Malicious software involves several stages, however the below stages mentioned can be considered as discrete and sequential steps over-simplifies the steps malware analysis process.

- Manual Code Reversing
- Interactive Behavior Analysis
- Static Properties Analysis
- Fully-Automated Analysis

Examining malicious software involves a variety of tasks, some simpler than others. These efforts can be grouped into stages based on the nature of the associated malware analysis techniques. Layered on top of each other, these stages form a pyramid that grows upwards in complexity. The closer you get to the top, the more burdensome the effort and the less common the skill set.

Fully-Automated Analysis

The easiest way to assess the nature of a suspicious file is to scan it using fully-automated tools, some of which are available as commercial products and some as free ones. These utilities are designed to quickly assess what the specimen might do if it ran on a system. They typically produce reports with details such as the registry keys used by the malicious program, its mutex values, file activity, network traffic, etc.

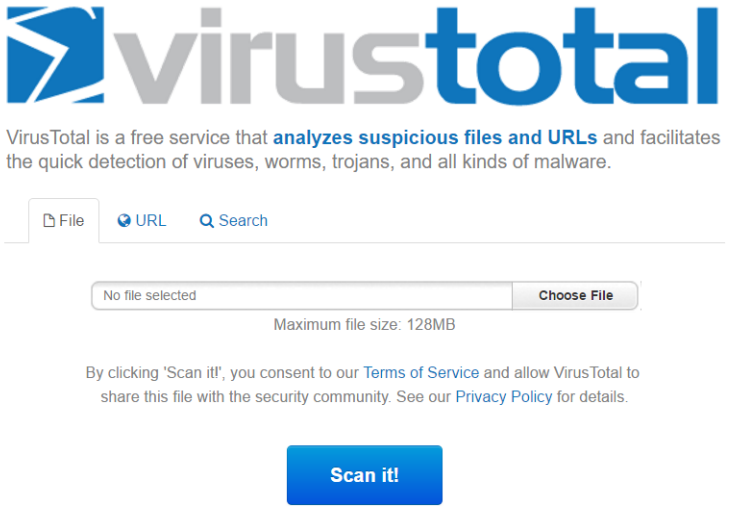
Fully-automated tools usually don't provide as much insight as a human analyst would obtain when examining the specimen in a more manual fashion. However, they contribute to the incident response process by rapidly handling vast amounts of malware, allowing the analyst (whose time is relatively expensive) to focus on the cases that truly require a human's attention.

Static Properties Analysis

An analyst interested in taking a closer look at the suspicious file might proceed by examining its static properties. Such details can be obtained relatively quickly, because they don't involve running the potentially malicious program. Static properties include the strings embedded into the file, header details, hashes, embedded resources, packer signatures, metadata such as the creation date, etc.

Looking at static properties can sometimes be sufficient for defining basic indicators of compromise. This process also helps determine whether the analyst should take closer look at the specimen using more comprehensive techniques and where to focus the subsequent steps. Analyzing static properties is useful as part of the incident triage effort.

VirusTotal is an example of an excellent online tool whose output includes the file's static properties.



Interactive Behavior Analysis

After using automated tools and examining static properties of the file, as well as taking into account the overall context of the investigation, the analyst might decide to take a closer look at

the specimen. This often entails infecting an isolated laboratory system with the malicious program to observe its behavior.

Behavioral analysis involves examining how sample runs in the lab to understand its registry, file system, process and network activities. Understanding how the program uses memory (e.g., performing memory forensics) can bring additional insights. This malware analysis stage is especially fruitful when the researcher interacts with the malicious program, rather than passively observing the specimen.

The analyst might observe that the specimen attempts to connect to a particular host, which is not accessible in the isolated lab. The researcher could mimic the system in the lab and repeat the experiment to see what the malicious program would do after it is able to connect. for example, if the specimen uses the host as a command and control (C2) server, the analyst may be able to learn about specimen by simulating the attacker's C2 activities. This approach to molding the lab to evoke additional behavioral characteristics applies to files, registry keys and other dependencies that the specimen might have.

Being able to exercise this level of control over the specimen in a properly orchestrated lab is what differentiates this stage from fully-automated analysis tasks. Interacting with malware in creative ways is more time-consuming and complicated than running fully-automated tools. It generally requires more skills than performing the earlier tasks in the pyramid.

Manual Code Reversing

Reverse-engineering the code that comprises the specimen can add valuable insights to the findings available after completing interactive behavior analysis. Some characteristics of the specimen are simply impractical to exercise and examine without examining the code. Insights that only manual code reversing can provide include:

- Decoding encrypted data stored or transferred by the sample;
- Determining the logic of the malicious program's domain generation algorithm;
- Understanding other capabilities of the sample that didn't exhibit themselves during behavior analysis.

Manual code reversing involves the use of a disassembler and a debugger, which could be aided by a decompiler and a variety of plugins and specialized tools that automate some aspects of these efforts. Memory forensics can assist at this stage of the pyramid as well.

Reversing code can take a lot of time and requires a skill set that is relatively rare. For this reason, many malware investigations don't dig into the code. However, knowing how to perform at least some code reversing steps greatly increases the analyst's view into the nature of the malicious program in a comp

Combining Malware Analysis Stages

The process of examining malicious software involves several stages, which could be listed in the order of increasing complexity and represented as a pyramid. However, viewing these stages as discrete and sequential steps over-simplifies the steps malware analysis process. In most cases, different types of analysis tasks are intertwined, with the insights gathered in one stage informing efforts conducted in another. Perhaps the stages could be represented by a "wash, rinse, repeat" cycle, that could only be interrupted when the analyst runs out of time.

Buffer Overflow

In computer security and programming, a buffer overflow, or buffer overrun, is an anomaly where a program, while writing data to a buffer, overruns the buffer's boundary and overwrites adjacent memory locations.

Buffers are areas of memory set aside to hold data, often while moving it from one section of a program to another, or between programs. Buffer overflows can often be triggered by malformed inputs; if one assumes all inputs will be smaller than a certain size and the buffer is created to be that size, if an anomalous transaction produces more data it could cause it to write past the end of the buffer. If this overwrites adjacent data or executable code, this may result in erratic program behavior, including memory access errors, incorrect results, and crashes.

Exploiting the behavior of a buffer overflow is a well-known security exploit. On many systems, the memory layout of a program, or the system as a whole, is well defined. By sending in data designed

to cause a buffer overflow, it is possible to write into areas known to hold executable code, and replace it with malicious code. Buffers are widespread in operating system (OS) code, so it is possible to make attacks that perform privilege escalation and gain unlimited access to the computer's resources. The famed Morris worm used this as one of its attack techniques.

Programming languages commonly associated with buffer overflows include C and C++, which provide no built-in protection against accessing or overwriting data in any part of memory and do not automatically check that data written to an array (the built-in buffer type) is within the boundaries of that array. Bounds checking can prevent buffer overflows, but requires additional code and processing time. Modern operating systems use a variety of techniques to combat malicious buffer overflows, notably by randomizing the layout of memory, or deliberately leaving space between buffers and looking for actions that write into those areas ("canaries").

Stack-based exploitation

Every Windows application uses parts of memory. The process memory contains 3 major components:

- code segment (instructions that the processor executes. The EIP keeps track of the next instruction)
- data segment (variables, dynamic buffers)
- stack segment (used to pass data/arguments to functions, and is used as space for variables. The stack starts (= the bottom of the stack) from the very end of the virtual memory of a page and grows down (to a lower address). a PUSH adds something to the top of the stack, POP will remove one item (4 bytes) from the stack and puts it in a register.

If you want to access the stack memory directly, you can use ESP (Stack Pointer), which points at the top (so the lowest memory address) of the stack.

- After a push, ESP will point to a lower memory address (address is decremented with the size of the data that is pushed onto the stack, which is 4 bytes in case of addresses/pointers). Decrements usually happen before the item is placed on the stack (depending on the implementation… if ESP already points at the next free location in the stack, the decrement happens after placing data on the stack)
- After a POP, ESP points to a higher address (address is incremented (by 4 bytes in case of addresses/pointers)). Increments happen after an item is removed from the stack.

When a function/subroutine is entered, a stack frame is created. This frame keeps the parameters of the parent procedure together and is used to pass arguments to the subrouting. The current location of the stack can be accessed via the stack pointer (ESP), the current base of the function is contained in the base pointer (EBP) (or frame pointer).

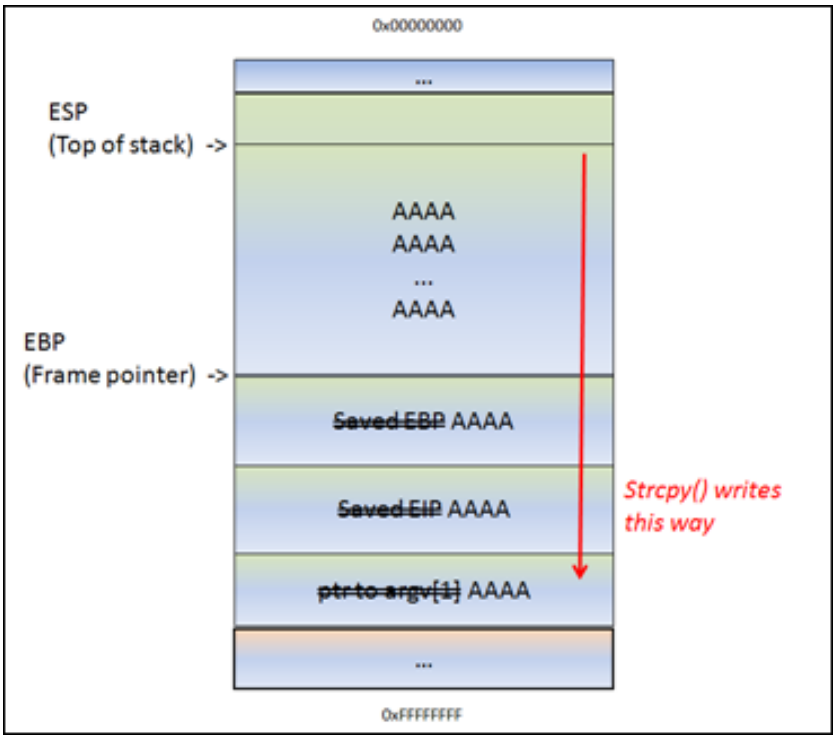
The CPU's general purpose registers (Intel, x86) are :

- EAX : accumulator : used for performing calculations, and used to store return values from function calls. Basic operations such as add, subtract, compare use this general-purpose register
- EBX : base (does not have anything to do with base pointer). It has no general purpose and can be used to store data.
- ECX : counter : used for iterations. ECX counts downward.
- EDX : data : this is an extension of the EAX register. It allows for more complex calculations (multiply, divide) by allowing extra data to be stored to facilitate those calculations.
- ESP : stack pointer
- EBP : base pointer
- ESI : source index : holds location of input data
- EDI : destination index : points to location of where result of data operation is stored
- EIP : instruction pointer

A technically inclined user may exploit stack-based buffer overflows to manipulate the program to their advantage in one of several ways:

1. By overwriting a local variable that is located near the vulnerable buffer on the stack, in order to change the behavior of the program
2. By overwriting the return address in a stack frame. Once the function returns, execution will resume at the return address as specified by the attacker - usually a user-input filled buffer
3. By overwriting a function pointer or exception handler, which is subsequently executed
4. By overwriting a local variable (or pointer) of a different stack frame, which will be used by the function which owns that frame later.

If the address of the user-supplied data used to effect the stack buffer overflow is unpredictable, exploiting a stack buffer overflow to cause remote code execution becomes much more difficult. One technique that can be used to exploit such a buffer overflow is called "trampolining". In that technique, an attacker will find a pointer to the vulnerable stack buffer, and compute the location of their shellcode relative to that pointer. Then, they will use the overwrite to jump to an instruction already in memory which will make a second jump, this time relative to the pointer; that second jump will branch execution into the shellcode. Suitable instructions are often present in large code.



Data Hiding

Data or information is very crucial to any organization or any individual person. None of us likes our conversation being overheard as it contains the potential of being misused. Same is the case with the data of any organization or of any person. The exchange of data among two potential parties must be in done in a secured method so as to avoid any tampering. Two types of threats exist during any information exchange. The unintended user who may try to overhear this conversation can either tamper with this information to change its original meaning or it can try to listen to the message with intention to decode it and use it to his/her advantage. Both these attacks violated the confidentiality and integrity of the message passed.

Cryptography

Cryptography is an art of transforming data into an unreadable format called cipher text. The receiver at other side, deciphers or decrypt the message into plain text. Cryptography provides

data confidentiality, data integrity, authentication and non-repudiation. Confidentiality is limiting access or placing restriction on certain types of information. Integrity is maintaining and assuring the accuracy of data being delivered, i.e, information contains no modification, deletion etc. Authentication ensures the identity of sender and receiver of the information.

Watermarking

A watermark is a recognizable image or pattern that is impressed onto paper, which provides evidence of its authenticity. Watermark appears as various shades of lightness/darkness when viewed in transmitted light. Watermarks are often seen as security features to banknotes, passports, postage stamps and other security papers. Digital watermarking is an extension of this concept in the digital world. A watermarking system's primary goal is to ensure robustness, i.e, it should be impossible to remove the watermark without tampering the original data

Steganography

Steganography is a practice of hiding/concealing the message, file, image within other message, file or image. The word steganography is of Greek origin and means "covered writing" or "concealed writing". In other words, it is the art and science of communicating in a way which hides the existence of the communication. The goal is to hide messages inside other harmless messages in a way that does not allow enemy to even detect that there is a second message present. Steganography focuses more on high security and capacity. Even small changes to stego medium can change its meaning. Steganography masks the sensitive data in any cover media like images, audio, video over the internet.

The first recorded uses of steganography can be traced back to 440 BC when Herodotus mentions two examples in his Histories. Histiaeus sent a message to his vassal, Aristagoras, by shaving the head of his most trusted servant, "marking" the message onto his scalp, then sending him on his way once his hair had regrown, with the instruction, "When thou art come to Miletus, bid Aristagoras shave thy head, and look thereon." Additionally, Demaratus sent a warning about a forthcoming attack to Greece by writing it directly

on the wooden backing of a wax tablet before applying its beeswax surface. Wax tablets were in common use then as reusable writing surfaces, sometimes used for shorthand.

Digital steganography, as stated before, is just a series of methods which hides information and files from view into other files and can have many beneficial and secure properties such as watermarking photographs to deter art theft, keeping sensitive data secure in innocuous files in case of unauthorized access or data theft, etc. But as any other tool in the world, intentionally and unintentionally, people may use this difficulty of detection in not such secure ways.

"Is your PC virus-free? Get it infected here!"

This was a real Google Ad last year. You may think that no one in his right state of mind would click this advert. But they do. Fortunately, this was only an experiment by Mikko Hypponen, who is Chief Research Officer at security firm F-Secure and only leads to a "Thank You" html page. During the six month period that this ad was online, 409 people either by mistake, out of curiosity or stupidity thought it was a good idea to click the link to "see what happens". This experiment was mentioned to show how some users willingly download viruses even if it says "Clicking this link will format your hard disk but you will see a dancing pig" let alone if the virus is hidden in an innocent attachment sent (seemingly) from a co-worker or a friend. (Anyone involved in computer security will know of the "Dancing pig problem"). The most common misuse of steganography is the hiding of malware into seemingly safe files such as pictures, audio and email attachments. This method is used to hide any type of malware ranging from viruses to worms from spyware to Trojans.

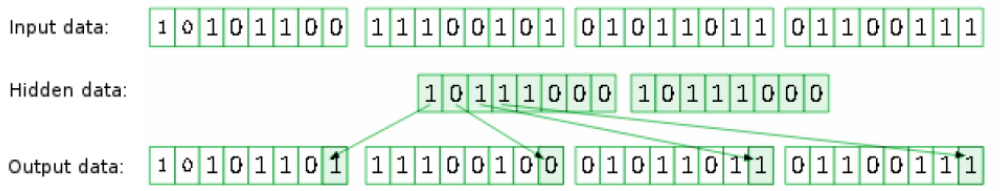
One of the simplest ways to hide malware is to use double extensions. A file would be named for example as "cutekitten.jpg.exe". When this is clicked, Windows will look only at the last part of the extension and therefore treats it as an executable. For an unprotected computer this method is particularly effective as this can be received as an attachment and, by default, Windows hides the last extensions of its files and therefore this is shown as a jpg file and can be overlooked and executed. An example was the Anna Kournikova virus which was sent via email as an attachment "AnnaKournikova.jpg.vbs". A similar technique is with URL links. These may be

fashioned to show that they are directed to a jpg, mp3 etc but when clicked, the user is redirected to an executable.

Macros embedded in Microsoft documents also fall under the steganography cap. These mini-programs are executed as soon as one opens the document and mostly spread by copying the email addresses in the address book and sending itself automatically by email. The Melissa virus is a famous example of this; it had a null payload but its damage came in the form of email servers congestion due to its high rate of spread. As stated before, text can be embedded in pictures. This may take the form of malicious code. Though harmless on its own, it can have a companion malware process which loads the program from the carrier picture. The main advantage is that in some systems, picture files are not scanned and the companion process will not have a virus signature.

While in the previous cases steganography was used to hide the malware to infect the system, it can also be used maliciously in reverse. A virus may be programmed to "hide" a user's important documents or files inside a file and ask for ransom for the password that will be used to decrypt the data back to its original state (hopefully). A macro famous for this was a variant of the Melissa virus mentioned before called Melissa.V. This macro made a backup of documents and destroyed random parts of the original. Then it requested a ransom of $100 to be transferred to an offshore account. Fortunately the owner of the account was tracked down and it was discovered that the macro wrote information in the Windows registry and with this, the documents could be retrieved.

A simple yet effective way of hiding data in an image for any purpose is to replace the least significant bit (LSB) of each byte in the cover with a single bit for the hidden message.



One big advantage of steganography, when compared to other cryptic processes, is that the hidden material is truly hidden from view. If everyone knows that there is some secret information somewhere, no matter how tight and secure it is, some cowboy will try to crack it and, given enough time and motivation, the data will be exposed.

Alternate data stream(ADS)– Windows File System

A file system is a part of the operating system on a volume and it determines how files are named, stored, and organized on basic or dynamic disks. A file system manages files and folders as well as the information required to locate and access these items by local and remote users. The Microsoft Windows Server 2003 operating system supports the NTFS and FAT file systems.

A relatively unknown compatibility feature of NTFS, Alternate Data Streams (ADS) provides hackers with a method of hiding root kits or hacker tools on a breached system and allows them to be executed without being detected by the systems administrator. When dealing with network security, administrators often times don't truly appreciate the lengths that a sophisticated hacker would go through to hide his tracks. Simple defacements and script kiddies aside, a sophisticated hacker with more focused goals looks to a perimeter system breach as an opportunity to progress further inside a network or to establish a new anonymous base from which other targets can be attacked. In order to achieve this task, a sophisticated hacker would need time and resources to install what is known as a root kit or hacker tools with which he can execute further attacks. With this, comes the need to hide the tools of his trade, and prevent detection by the systems administrator of the various hacking applications that he might be executing on the breached system.

One popular method used in Windows Systems is the use of Alternate Data Streams (ADS). A relatively unknown compatibility feature of NTFS, ADS is the ability to fork file data into existing files without affecting their functionality, size, or display to traditional file browsing utilities like dir or Windows Explorer. Found in all version of NTFS, ADS capabilities where originally conceived to allow for compatibility with the Macintosh Hierarchical File System, HFS; where file information is sometimes forked into separate resources. Alternate Data Streams have come to be used

legitimately by a variety of programs, including native Windows operating system to store file information such as attributes and temporary storage.

Amazingly enough, Alternate Data Streams are extremely easy to make and require little or no skill on the part o the hacker. Common DOS commands like “type” are used to create an ADS. These commands are used in conjunction with a redirect [>] and colon [:] to fork one file into another.

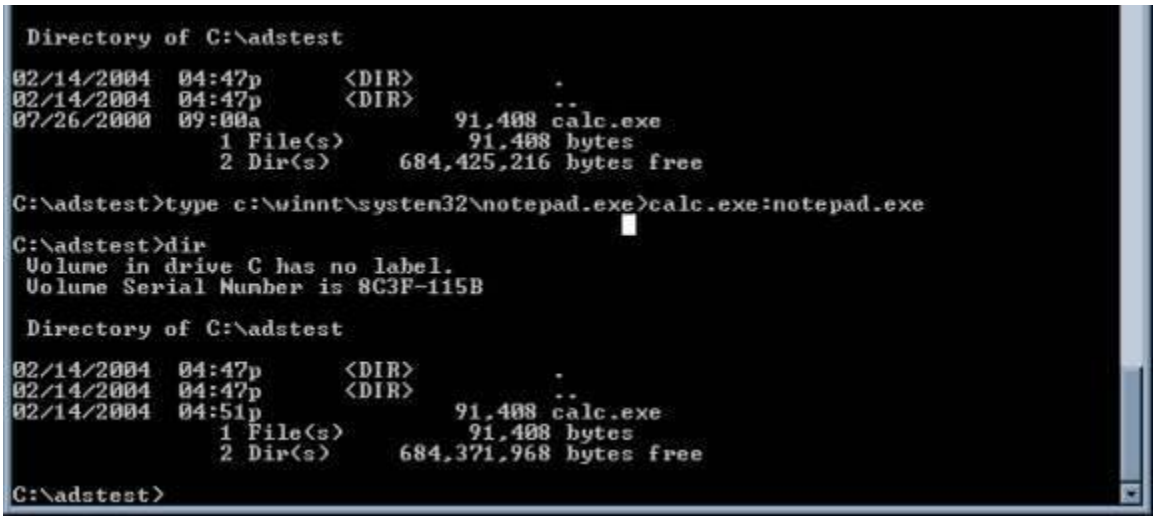
For instance: the command

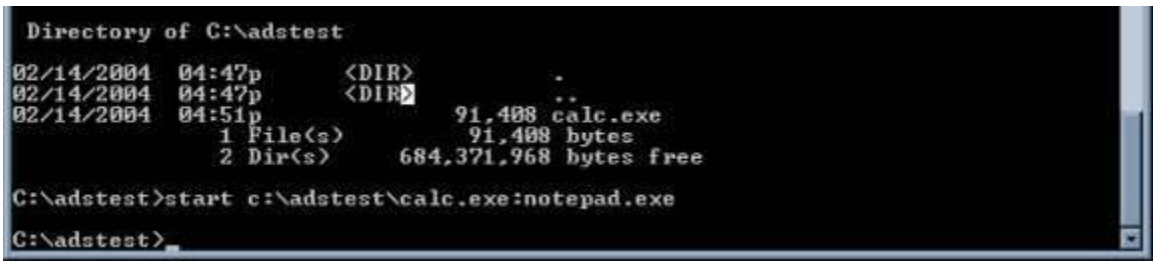
“type c:\anyfile.exe > c:\winnt\system32\calc.exe:anyfile.exe”

will fork the common windows calculator program with an ADS “anyfile.exe.”

Alarmingly files with an ADS are almost impossible to detect using native file browsing techniques like command line or windows explorer. In our example, the file size of calc.exe will show as the original size of 90k regardless of the size of the ADS anyfile.exe. The only indication that the file was changed is the modification time stamp, which can be relatively innocuous. Once injected, the ADS can be executed by using traditional commands like type, or start or be scripted inside typical scripting languages like VB or Perl. When launched, the ADS executable will appear to run as the original file - looking undetectable to process viewers like Windows Task Manager. Using this method, it is not only possible to hide a file, but to also hide the execution of an illegitimate process.

Unfortunately, it is virtually impossible to natively protect your system against ADS hidden files if you use NTFS. The use of Alternate Data Streams is not a feature that can be disabled and currently there is no way to limit this capability against files that the user already has access to. Freeware programs like lads.exe by Frank Heyne (www.heysoft.de) and crucialADS by CrucialSecurity can be used to manually audit your files for the presence of Alternate Data Streams. Alternatively, the action of moving a file into another file system that doesn’t support ADS will automatically destroy any Alternate Data Streams.





Ultimately only a third party file checksum application can effectively maintain the integrity of an NTFS partition against unauthorized Alternate Data Streams. Recently dubbed as host based "Intrusion Prevention Systems" or "Intrusion Detection Systems", third party security applications like eTrust Access Control from Computer Associates have been used for years in high-end government networks to verify the integrity of files used in the most secure environments. In addition to a heightened level of auditing and access control, these applications typically create an MD5 hashed database of file checksums that are used to validate a file's trustworthiness. File injection techniques like Alternate Data Streams trigger an action by which the file is deemed untrusted and therefore prevented from executing or better yet, prevented from being changed in the first place.

Threat actors

Over the past few years, cyber security has made its way onto every organization's radar. Hardly a week goes by without another high-profile breach, and with each new headline cyber security budgets across the globe are growing ever larger.

But unfortunately, simply spending more money isn't enough. To avoid the cost and embarrassment of a data breach, you'll need to understand your adversaries.

Most threat actors fall within four main groups, each with their own favorite tactics, techniques, and procedures (TTPs). By gaining a deeper understanding of threat actors, you'll be able to assign your cyber security budget to fund the right activities.

Cyber Criminals, Organized and Otherwise

When thinking about cyber criminals, many imagine some nerdy hacker sitting in his mom's basement eating potato chips. This couldn't be further from the truth. These days cybercrime is far more organized than ever before, and last year it even overtook the drug trade to become the most profitable illegal industry. To give you some idea of scale, it's estimated that victims in the U.S. paid over $24 million in 2015 to groups using ransomware Trojans, and that's just one attack vector. These groups are well equipped, well-funded, and they have the tools and knowledge they need to get the job done. But to really understand cyber criminals, you just need to know one thing: their motives. Overwhelmingly, cyber criminals are interested in money. Either they'll use ransomware to extort money from you, or they'll steal data that can be sold via dark web markets.

Common TTP

Right now, cyber criminals are all about mass phishing campaigns. It's low cost, easy to pull off, and promises a truly staggering return on investment. Sure, spear phishing is still a big concern, and it's much harder to defend against, but for pure bang-for-your-cyber-criminal-buck nothing beats a good mass phish. Typically these campaigns are used to deliver malware payloads (often ransomware), and emails usually include a strong social engineering component. For instance,

recipients are often asked to open or forward attachments such as office documents which in turn activate malicious software when opened.

How to Defend Against It

Keep in mind the cyber criminal's focus on profit. If they can't convince you to pay a ransom or sell your data, you're useless to them. Since phishing is the current weapon of choice for cyber criminals, the best defenses are email filtering and authentication systems. By scanning all incoming and outgoing email for suspicious content (e.g., executable files, "spammy" language, or similarity to previously intercepted emails), you'll be able to block and quarantine the vast majority of malicious spam. High-quality threat intelligence is extremely beneficial here, as it can be used to constantly improve spam filters and prevent the latest phishing emails from finding their mark. Equally, some phishing emails originate from domains and IPs that are easily blocked. Using technologies such as Sender Policy Framework (SPF), Domain Message Authentication Reporting and Conformance (DMARC), and DomainKeys Identified Mail (DKIM) will help you avoid a lot of headaches.

Hacktivists

Unlike cyber criminals, hacktivists are generally not motivated by money. Instead, they have a burning rage inside them that for whatever reason has been directed at you. They often work alone, making their attacks extremely difficult to predict or even respond to quickly — but don't underestimate them. Many hackers, ethical or not, are actively involved with the cyber security industry in some capacity. But whether they're a network administrator, a mid-level IT guy, or even a college student, there's no way of knowing in advance who they are or when they'll strike. Of course, it's difficult to really pin down a hacktivist's motives in advance, but it is possible to predict their actions. Since they aren't interested in money, hacktivists are usually in the business of cyber vandalism. If they do aim to steal your data, it's probably because they expect to find something incriminating, or simply wish to cause you embarrassment.

Common TTP

According to Control Risks, hacktivists overwhelmingly favor attacking websites. Since its website is often the most publicly facing aspect of an organization, this makes perfect sense. But how do they do it? Well, for many years now, DDoS (distributed denial of service) attacks have been a firm favorite. To initiate a DDoS attack, a hacktivist must first take control of a large number (usually thousands or tens of thousands) of computers, which they typically achieve by using malware spam campaigns.

Once they have control, the hacktivist will use his "botnet" to repeatedly send simple requests (e.g., viewing a webpage) to a specific website over and over again. The amount of traffic generated by a DDoS attack can be truly staggering, and often leads to site crashes and large hosting bills for the website owner.

How to Defend Against It

Defending against DDoS attacks isn't easy. First, you'll need your incident response planning to be spot on. Not only that, you'll need to identify the signs of DDoS attacks early on, and give yourself the best possible chance to mitigate the attack before it reaches its inevitable conclusion. Finally, there are a number of DD0S mitigation products and services on the market, so give serious consideration to investing in one of these.

State-Sponsored Attackers

In recent years, we've all heard a lot about state-sponsored attacks and cyber espionage. In reality state-sponsored attacks are far less common than cyber-crime and hacktivism, but they are nonetheless a real and concerning trend. Unsurprisingly, state-sponsored attackers aren't usually interested in your money. At least, not directly. Instead, they want your data, and that means gaining sustained access to your IT infrastructure. If your organization operates in a particularly sensitive market where proprietary data is jealously guarded (e.g., technology, pharmaceuticals, or finance), you're at a greater risk of gaining the attentions of a state-sponsored hacking group.

Common TTP

Since state-sponsored attackers need long-term access to your IT infrastructure, their preferred TTP is known as the advanced persistent threat (APT). Unfortunately, this term is less precise than you might hope. In essence, because so much is on the line, state-sponsored groups will often work on multiple attack vectors simultaneously, even if they already have access to your infrastructure. In this way, they can collect sensitive data over a long-time period, rather than simply performing a smash-and-grab operation. Sadly, although the average time to detect a breach fell substantially last year, it's still in the region of five months. Needless to say, nobody wants a state-sponsored hacking group intercepting their private data for even a day, so five months is clearly too long.

How to Defend Against It

Since APTs make use of multiple attack vectors, there's no single security silver bullet to keep your organization safe. Instead, you'll need to build a strong, consistent, and ongoing security program that includes both the fundamentals (e.g., vulnerability and patch management) and the more advanced (threat intelligence). Effective cyber security is a marathon, not a sprint, so if you're starting from scratch you certainly won't be able to do everything. Focus on building up your cyber security program one piece at a time, and always look for ways to improve. Ultimately, even with state-sponsored groups, if you can make their job really difficult, there's a good chance they'll go elsewhere in search of easier targets.

The Insider Threat

Don't be fooled into thinking that all insider threats are the same. Some are simply normal employees who want to be helpful and end up giving away sensitive data to the wrong person. Others feel maligned by their organization, and want to get their own back. Still more are real user accounts which have been compromised by an external attacker. But whatever their circumstances or motives, insider threats are dangerous, and often hard to spot. They may aim to vandalize assets as a form of revenge, steal proprietary assets for resale on the dark web, or simply send sensitive data to anybody who asks. And the hard part, of course, is distinguishing these actions from all the legitimate activity that occurs every day on your network.

Common TTP

Although insiders do sometimes commit acts of vandalism, information is usually their target. Insider threats have led to some of the largest data breaches in history, so protecting confidential data should be your organization's primary concern.

How to Defend Against It

First off, your well-meaning employees should be at the top of your list. Most people want to be helpful, and this trait can be (and often is) abused by hackers to achieve their goals. Security awareness training is an absolute must here, because after all, you may have disgruntled employees, but you'll always have gullible employees. For compromised or malicious insiders, a different tactic is needed. Since they'll be looking for sensitive data, using honeypots in combination with user behavior analytics will enable to you identify those users who are actively searching for data they shouldn't have. And once you've identified them, you can follow their behavior more closely, and quickly put together the evidence you need to confront them.

Act of God

Natural disasters, such as earthquakes, floods and hurricanes, can damage your computer. Fires, extreme temperatures and lightning strikes can cause major physical damage and lead to loss of data.

Fire can cause serious damage to your computer. Even if the computer does not actually catch fire, the heat can be enough to damage delicate components. Smoke can damage the CPU fan, causing damage to the CPU due to overheating.

Many components of your computer are designed to operate within a specific range of temperatures. Some components may start to malfunction if they experience excessive heat or cold and you may need to replace them. If your computer has been exposed to extreme temperatures you should let it return to room temperature before starting it again.

Lightning strikes carry a huge electrical charge which can cause a surge. A surge or spike is a sudden increase in voltage, which can cause damage to some components of your computer,

including the motherboard, CPU and memory. Modems and routers are often damaged by lightning strikes.

How to Defend Against It

There are a number of fairly simple steps you can take to protect your computer and the data stored on it against natural threats:

- Make a backup of the data stored on your computer. This means that you will have additional copies of your data which will let you recover from any data loss. You should ideally keep a copy of any important data in a physically separate location. USB drives can be useful for keeping backups. Nowadays people are making increased use of online or cloud-based backup services.
- Install your computer in a location where it is unlikely to be damaged due to natural factors.
- Install an Uninterruptible Power Supply (UPS) to provide battery backup in the event of a power cut. This prevents damage caused by the sudden shutting down of your computer. During major electrical storms, you should turn off your computer and unplug it from the power socket to avoid damage.

Industrial Espionage in Cyberspace

When you hear the word espionage, perhaps you conjure up a number of exciting and glamorous images. Perhaps you have visions of a well-dressed man who drinks martinis, shaken but not stirred, traveling to glamorous locations with equally glamorous travel companions. Or perhaps you envision some exciting covert operation with high-speed car chases and guns blazing in faraway exotic lands. Contrary to popular media portrayals, espionage is often much less exciting than those visions. The ultimate goal of espionage is to obtain information that would not otherwise be made available. Generally, espionage is best done with as little fanfare as possible. Blazing gun battles and glamorous locations tend to be the antithesis of intelligence gathering.

Rather, information is the goal. If possible, it is best to obtain that information without the target organization even realizing that its information has been compromised.

Many people assume that such spying is only engaged in by governments, intelligence agencies, and nefarious international organizations, such as Al Qaida or ISIS. While those entities do indeed engage in espionage, they are certainly not the only organizations that do so. The aforementioned organizations desire to acquire information for political and military goals. However, economic goals are also dependent on accurate and often sensitive data. With billions of dollars at stake, private companies can become engaged in industrial espionage as either a target or a perpetrator. What company would not like to know exactly what its competitor is doing? In fact, corporate or economic espionage is on the rise.

Corporate or economic espionage is a growing problem, but it can be difficult to accurately assess just how great a problem it is. Companies that perpetrate corporate espionage do not share the fact that they do it, for obvious reasons. Companies that are victims of such espionage often do not wish to reveal that fact either. Revealing that their security was compromised could have a negative impact on their stock value. It is also possible, in certain cases, that such a breach of security might open the company to liability claims from customers whose data may have been compromised. For these reasons, companies often are hesitant to disclose any industrial espionage activities. Because you will want to protect yourself and your company, it is important that you learn about espionage methods and protections.

Information as an Asset

Many people are used to viewing tangible objects as assets but have difficulty appreciating how mere information can be a real asset. Companies spend billions of dollars every year on research and development. The discovered information is worth at least the amount of resources taken to derive the information plus the economic gain produced by the information. For example, if a company spends $200,000 researching a process that will in turn generate $1 million in

revenue, then that data is worth at least $1.2 million. You can think of this economic gain as a simple equation:

```
VI (value of information) = C (cost to produce) + VG (value gained)
```

Real-World Examples of Industrial Espionage

Example 1: Houston Astros

In 2015 the Houston Astros baseball team's scouting and team information database was stolen. It is alleged that it was stolen by members of the St. Louis Cardinals. The Houston Astros have a proprietary internal computer system they named Ground Control. It has notes on players and potential trading of players. The Astros general manager, Jeff Luhnow , had previously worked for the Cardinals, and when he came to work for the Astros, he also brought along some of his staff. Initial reports are that either Mr. Luhnow or one of his staff used a password similar to what he had used with the Cardinals. This allowed someone associated with the Cardinals to guess the password and access the Houston Astros database.

Example 2: University Trade Secrets

In May 2015, Professor Hao Zhang of Tianjin University and five other individuals were arrested and charged with stealing trade secrets for use by universities controlled by the Chinese government. The secrets stolen included research and development on thin-film bulk acoustic resonator (FBAR) technology. FBAR is essentially a device that has material located between two electrodes and acoustically isolated from the medium it is in. This is commonly used as a radio frequency filter in cell phones.

Example 3: VIA Technology

VIA Technology actually provides two examples of industrial espionage. In the first instance, the chief executive officer (CEO) of the firm, which was based in Taipei, was indicted for copyright infringement for allegedly stealing technology from one of his own customers, a

networking company called D-Link (Network World Fusion, 2003). According to the allegations, VIA engineer Jeremy Chang left VIA to work for D-Link. For several months while at D-Link, Chang continued to receive a paycheck from VIA. Then he promptly resigned from D-Link and returned to VIA. Once Chang rejoined VIA, a D-Link document that detailed one of its simulation programs for testing integrated circuits was posted to an FTP server owned by VIA. The prosecutors allege that Chang continued to receive a check from VIA because he had never really resigned. They allege that Chang was in fact a "plant" sent to D-Link to acquire D-Link's technology for VIA. VIA maintains that his continuation to receive a check was simply an oversight, and Chang denies that he posted the document in question. Whatever the truth of the case, it should make any employer think twice about hiring decisions and nondisclosure agreements. To make matters worse for VIA, another company accused VIA of stealing code for its optical readers. In both cases, the story of the possible theft of technology alone has had a negative impact on the stock value of both companies.

Example 4: General Motors

In 1993, General Motors (GM) and one of its partners began to investigate a former executive, Inaki Lopez . GM alleged that Lopez and seven other former GM employees had transferred GM proprietary information to Volkswagen (VW) in Germany via GM's own network (Brinks et al., 2003). The information allegedly stolen included component price data, proprietary construction plans, internal cost calculations, and a purchasing list. In 1996, GM followed up the ongoing criminal investigation with civil litigation against Lopez, VW, and the other employees. In November 1996, GM expanded its legal battle by invoking the various Racketeer Influenced and Corrupt Organizations Act (RICO) statutes, originally intended to be used against organized crime conspiracies (*Economist,* 1996). By May 2000, a federal grand jury indicted Lopez on six counts related to fraud and racketeering. As of this writing, the case is not resolved (*USA Today,* 2000). At the time Lopez was indicted, he was residing in Spain, and the U.S. Justice Department was negotiating for his extradition. Thus, you can see that corporate espionage is neither new nor restricted to technology companies.

Example 5: Bloomberg, Inc.

According to the American Bar Association Journal (2003), in August 2003, Oleg Zezev , a 29-year-old PC technician from Kazakhstan, broke into the Bloomberg Inc. computer system and used the alias Alex to obtain information and then blackmail the firm. Zezev entered Bloomberg's computer system and accessed various accounts, including Michael Bloomberg 's (CEO and founder of Bloomberg L.P.) personal account as well as accounts for other Bloomberg employees and customers. Zezev copied information from these accounts, including email inbox screens, Michael Bloomberg's credit card numbers, and screens relating to the internal functions of Bloomberg. He also copied internal information that was only accessible by Bloomberg employees. Zezev then threatened to expose the data he had stolen to the public and, in essence, tell everyone exactly how he had broken into Bloomberg's network unless he received $200,000. After deliberating for less than six hours, the jury in the U.S. District Court in Manhattan found the perpetrator guilty of all four charges: conspiracy, attempted extortion, sending threatening electronic messages, and computer intrusion. Although this is not industrial espionage in the classic sense, it does illustrate the compromising situations in which a company and its employees can be placed when security is breached.

Example 6: Interactive Television Technologies, Inc.

On August 13, 1998, someone broke into the computer systems of Interactive Television Technologies, Inc. and stole the data for a project the company was working on (Secure Telecom, 1998). That project involved four years of intense research and a substantial financial investment. The product was to be a way whereby anyone with a television could have Internet access via the Web. This product, code named "Butler," would have been worth a substantial amount to its inventors. However, with all the research material stolen, it was only a matter of time before several other companies came out with competing products, thus preventing Interactive Television Technologies from pursuing a patent. To date, no arrests have been made and no leads are available in this case. This situation was a case of very skillful hackers breaking into a computer system and taking exactly what they needed. One can only speculate about their motives. They

may well have sold the research data to competitors of Interactive Television Technologies, or they may have simply put the data out in the open via the Internet. Whatever the motives or profits for the perpetrators, the outcome for the victim company was catastrophic.

Low-Tech Industrial Espionage

Corporate espionage can occur without the benefit of computers or the Internet. Disgruntled former (or current) employees can copy sensitive documents, divulge corporate strategies and plans, or perhaps reveal sensitive information. In fact, whether the method used is technological or not, disgruntled employees are the single greatest security risk to any organization. A corporate spy need not hack into a system in order to obtain sensitive and confidential information if an employee is willing to simply hand over the information. Just as with military and political espionage, the motives for the employee to divulge the information vary. Some engage in such acts for obvious financial gains. Others may elect to reveal company secrets merely because they are angry over some injustice (real or imagined). Whatever the motive, any organization has to be cognizant of the fact that it has any number of employees who may be unhappy with some situation and have the potential to divulge confidential information.

Certainly, one can obtain information without the benefit of modern technology; however, computer technology (and various computer-related tactics) can certainly assist in corporate espionage, even if only in a peripheral manner. Some incidents of industrial espionage are conducted with technology that requires little skill on the part of the perpetrator. This technology can include using universal serial bus (USB) flash drives, compact discs (CDs), or other portable media to take information out of the organization. Even disgruntled employees who wish to undermine the company or make a profit for themselves will find it easier to burn a wealth of data onto a CD and carry that out in their coat pocket rather than attempt to photocopy thousands of documents and smuggle them out. And the new USB flash drives, smaller than your average key chain, are a dream come true for corporate spies. These drives can plug into any USB port and store a tremendous amount of data. As of this writing, one can easily purchase small portable devices capable of holding 2 terabytes or more of data.

While information can be taken from your company without overt hacking of the system, you should keep in mind that if your system is unsecure, it is entirely possible that an outside party would compromise your system and obtain that information without an employee as an accomplice. In addition to these methods, there are other low-tech, or virtually "no-tech," methods used to extract information. The first and most obvious use of social engineering in industrial espionage is in direct conversation in which the perpetrator attempts to get the targeted employee to reveal sensitive data. Employees will often inadvertently divulge information to a supplier, vendor, or salesperson without thinking the information is important or that it could be given to anyone. This involves simply trying to get the target to talk more than they should. In 2009, there was a widely-publicized case of a Russian spy ring working in the United States. One of their tactics was simply to befriend key employees in target organizations and, through ongoing conversations, slowly elicit key data. Another interesting way of using social engineering would be via email. In very large organizations, one cannot know every member. This loophole allows the clever industrial spy to send an email message claiming to come from some other department and perhaps simply asking for sensitive data. A corporate spy might, for example, forge an email to appear to be coming from the legal office of the target company requesting an executive summary of some research project.

Spyware Used in Industrial Espionage

Clearly, any software that can monitor activities on a computer can be used in industrial espionage. Security IT World, an online e-zine, featured an article in its October 2003 issue that dealt with the fact that monitoring a computer is an easy thing to do in the twenty-first century. The problem still persists to this day, with many security experts stating that spyware is at least as widespread as viruses. One method to accomplish monitoring is via spyware, "Malware." Clearly, software or hardware that logs key strokes or takes screenshots would be most advantageous to the industrial spy. The application of this type of software to espionage is obvious. A spy could get screenshots of sensitive documents, capture logon information for databases, or in fact capture a

sensitive document as it is being typed. Any of these methods would give a spy unfettered access to all data that is processed on a machine that contains spyware.

Exploit Kits

Before we talk about exploit kits, let's start with exploits. In computer security, an exploit is an object that causes a program to behave in an unexpected manner. The object is usually something that the program is unable to deal with – for example, a string of characters that does not fit an expected pattern, or a series of commands that the program is unable to correctly execute. When an exploit forces the program to behave unexpectedly, an attacker can take advantage of the disruption to perform other, usually malicious, actions that would not normally be permitted. For example, an attacker might exploit one program on a computer in such a way that a second program is silently installed without authorization from the user – an action that would normally be detected and blocked by the operating system.

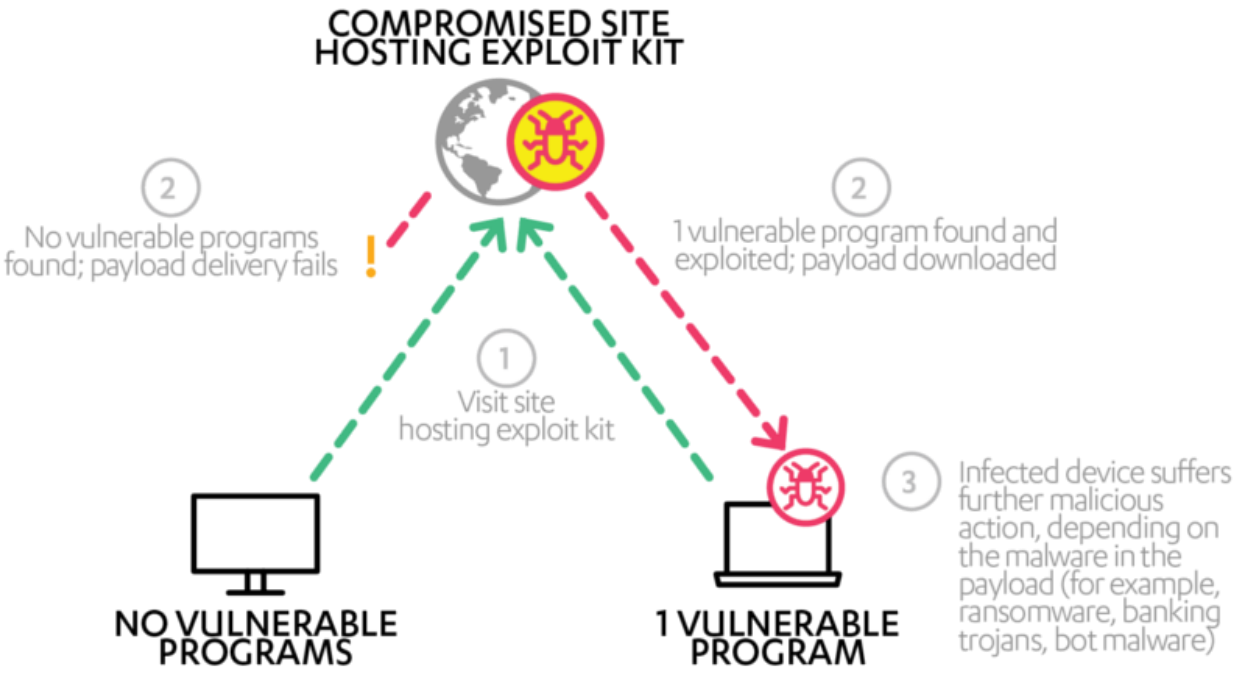
When a program is unable to deal with an exploit because of an underlying flaw or loophole in its coding or implementation, the flaw is known as a vulnerability. Vulnerabilities can be found in any type of software, from simple macro scripts that run within a computer program, to the software itself, to the operating system that runs it and even on the 'firmware' that controls the physical components of a user's computer or mobile device. For an exploit to be a danger however, an attacker must have some way to deliver it to the vulnerable program. For some vulnerabilities, this requires the attacker to have physical access to the targeted computer or mobile device, which obviously limits an attacker's opportunities. Far more dangerous is when an attacker can leverage a vulnerability from a distance, most commonly over the Internet – and that's where exploit kits come in.

An exploit kit is basically a utility program or toolkit that can deliver an exploit to its corresponding target program. If the exploit is successful, the kit can then deliver a malicious payload to the compromised computer or mobile device. If you think of a single exploit as being an 'arrow' that

can only hit one particular 'sweet spot' on a target, then an exploit kit is the 'bow' that can launch an entire quiversful of arrows at any target that happens to be within range.

In order to get targets to attack, exploit kit operators will typically host their kits on websites, which may be either maliciously crafted websites, or legitimate ones that have been compromised. The kits can then silently probe the computers or mobile devices of any visitors to the site. In some cases, attackers may increase the flow of potential victims to the exploit kit by using some form of web traffic hijacking to redirect more visitors to the poisoned website. For example, websites might be hacked in order to quietly redirect users to the site hosting the exploit kit.

If a visitor's machine is found to be vulnerable to the exploit, the kit then downloads a payload onto the victim (essentially, a *drive-by download* attack). The payload can be tailored according to the exploit kit operator's wishes, but typically include downloading such malware as ransomware, botnet-related components and banking-trojans.



Most exploit kits can also be updated by their creators or controllers (not always the same party) to add new exploits, allowing them to target any new vulnerabilities found without much fuss. For example, when the Hacking Team data breach occurred in early 2015, exploit code that was detailed in the exposed data was quickly added to various exploit kits.

Why are exploit kits a concern?

Exploit kits have become one of the more prevalent threats online today because they are essentially crimeware - specialized utility programs that are offered for sale (or rent) by their creators to interested third parties in various crime-oriented forums. On the modern Internet, they are actively being used by less technically-savvy attackers as a relatively easy way to attack and infect a large number of users.

Unlike previous forms of malware, which tended to be operated by only a small number of attackers (or even just by their creators), crimeware can be used by anyone who is able to purchase the 'product', making the potential pool of attackers much larger. Since new exploits can be simply added to an exploit kit's arsenal, attackers can also keep using the same tool (with the appropriate updates) over a longer period, in comparison to a similar, more single-focused malware that tends to have a shorter shelf life.

There are multiple exploit kits in the wild, though what vulnerabilities they target and how prevalent they are is quite variable. Today, some of the more well-known exploit kits include Angler, Magnitude, Nuclear and Neutrino (though this may not be true for long as fortunes change quite quickly in the exploit kit underworld, as can be seen when the Blackhole kit went from dominant to negligible after its operator was arrested).

How do I protect my device against exploit kits?

A successful exploit-based attack allows an attacker to gain a toehold on a computer or device, which they can then use to launch a longer chain of further intrusion and mischief. Intercepting

an exploit before it completes is therefore an efficient way to stop a wider-scale attack before it can really get underway.

Since exploits are most commonly delivered by exploit kits today, we'll concentrate on what users can do to evade or block attacks launched from an exploit kit. An exploit kits' attack can only proceed if it is having two consecutive opportunities to attack your computer or device:

- Opportunity 1: You visit (or are redirected to) a website hosting an exploit kit
- Opportunity 2: The vulnerability their exploits leverage is unpatched and undefended

Both opportunities must be present for an attack from an exploit kit to succeed: a vulnerable machine that never encounters the exploit kit can't be attacked, any more than one that is exposed to a kit but doesn't have any vulnerable programs installed. So, to evade attacks from exploit kits, a user would need to avoid providing at least one (and preferably both) of these 'openings' for attack.

There are various steps you can take when surfing online to avoid encountering exploit kits. For example, website security rating services help users avoid known malicious or compromised websites, while script blocking software and antivirus programs prevent malware from redirecting the browser to an unsolicited site. More concretely, users can render exploits pointless by removing their intended target and closing the flaw in a vulnerable program with a security patch issued by the program's vendor. Users are strongly urged to install security patches for any software installed on their computers or devices as soon as they are released.

Cyber Crime/Malware as a Service

Malware-as-a-Service is a prosperous business run on the black market that offers an array of services and isn’t just limited to malware or bits of code. And you don’t have to be a computer expert either. Anyone can purchase code that will cause harm to a person’s computers or even hold it for ransom. But once purchased, what are you going to do with it? How will investing in this piece of malware return a profit? There’s still the challenge of getting it out there, getting your

potential victims to run the payload for the newly purchased malware on their computer. And most importantly, cashing out on the investment. This is where the entire business model of Malware-as-a-Service comes into play.It's all offered in the cyber black market and functions no different than the global markets we hear of. Due to its low key nature, it's difficult to say exactly how much money is generated from Malware-as-a-Service in this market. But it would be no surprise if it stretched up into the billions. In this market it's possible to purchase all the necessary pieces to make it as easy as possible for the investors to profit.

www.easycybersec.com

Phone: +91 – 9597723169
WhatsApp: +91 – 9791956182

Malware As A Service

The Elite Group

The Distributors

The Money Mules

- First level: The highly skilled elite programmers or engineers who write malware, develop exploits, and are general researchers. This can be an individual or individuals working together.
- Second level: Here are the spammers, botnet owners, distributors, hosted system providers. These people are also skilled, but not always elite. This is where the distribution is handled
- Third level: The money mules, treasurers, financial data providers.

These three levels fall under the umbrella of Malware-as-a-Service that can be sold and purchased as an entire package or individual services by a vendor.

The hottest growth segment in Cybercrime-as-a-service is ransomware, a technique that uses encryption technology to deny victims access to their own data until they pay up. The number of ransomware domains tracked in the DNS Threat Index has increased 35 times from its baseline value. Ransomware has hit the big time — not just in the sheer number of malicious websites involved, but also in the scale of attacks and the nature of the targets. Ransomware used to be associated with small-scale attacks aimed largely at consumers or small businesses. Now, enterprise-strength ransomware attacks can target even the largest organizations.

The individuals involved aren't always strictly black hat. There are also grey hat hackers, otherwise known as freelancers who are simply looking to make a profit. A programmer can sell a zero-day exploit to the vendor of a software as a bounty. However, that same exploit might be able to fetch a far greater profit if sold on the black market. A perfect example of this is Facebook, who offers a minimum of $500 for anyone who can hack their site. With over 700 million users, a Facebook exploit can sell for a pretty hefty price in the black market. As malware becomes more profitable this type of business model will continue to grow.

Threats in Internet of things

The Internet of Things (IoT) is a mass of billions of connected devices from cars to wireless wearable products. Cisco's Internet Business Solutions Group estimated 12.5 billion connected devices in existence globally as of 2010 with that number doubling to 25 billion by 2015.

In-Car WiFi

Once tallied, 2013 connected car revenues should reach $21.7 billion, according to analysts from Visiongain, LTD, with 2014 revenues climbing even further. As of the New Year, Ford and GM will increasingly offer in-car Wi-Fi, turning cars into mobile hotspots and connecting passengers' smartphones, tablets and other devices to the Internet, according to John Pescatore, Director of Emerging Trends, the SANS Institute.

But, in-car WiFi has the same security vulnerabilities as traditional Wi-Fi hotspots. Without the firewalls present in conjunction with small business Wi-Fi installations, in-car devices and data will be at risk. Once inside the network, an attacker can spoof (pose as) the car, connect to outside data sources such as OnStar servers and collect the owner's PII such as credit card data, explains Pescatore. That is just one example. Only the imagination can limit the kinds of attacks that become possible when a hacker owns in-car Wi-Fi, passengers' devices and the car's identity (via spoofing).

mHealth Applications / Mobile Medical Devices

"The market for wearable wireless devices across sports, fitness and mHealth will grow from 42 million devices in 2013 to 171 million in 2018," says Jonathan Collins, Lead Analyst, ABI Research. As of 2014, hackers will increasingly attack mobile medical devices running Windows, including pacemakers, according to Rodney Joffe, Senior Technologist, Neustar. Traditional manufacturers use proprietary embedded systems that are hard to hack due to their closed source code and restrictions. But, non-traditional device manufacturers often use a form of Windows.

"Windows is very popular for those devices because it is cheap, ubiquitous and well-known among programmers," explains Joffe. But, unlike Windows on a desktop computer, there is no patching mechanism for Windows on these devices, according to Joffe. The more these devices connect to the Internet through wireless frequencies such as WiFi, the more viruses will spread among them.

Wearable Devices, Google Glass

The global wearable technology market will reach $4.6 billion in value in 2013, according to Visiongain, LTD, and continue to rise in 2014. In that market, devices such as Google Glass are a major attack vector because they automatically connect to the Internet. And, these devices have very few if any security solutions on them. Hacking Google Glass provides attackers with confidential corporate information and intellectual property. An organization may not know what kinds of data or how much a wearer absorbs using Google Glass as they move through offices and other environments in the enterprise. A hacker could copy that audio and video.

"Every organization should write policies for wearable devices that limit where these things can be used, when they can be used, and what their acceptable use is," Irvine says.

Drones (unmanned aircraft) for domestic (non-military) use

In February of 2012, the Congress established the FAA Modernization and Reform Act with numerous provisions for unmanned aircraft with the general thrust that the FAA will speed the inclusion of UAVs/drones in the national airspace system in three years' time (by 2015). "Drones will be prevalent across the country five years from now," says Erik Cabetas, Managing Partner, Include Security, LLC. CSOs should start to plan for drone security measures now.

"Because drones rely on vulnerable telemetry signals, attackers can leverage them using any of the classic attacks including buffer overruns, format strings, SQL injections and authentication bypasses that exist in drone firmware," explains Cabetas.

Examples of successful attacks on drones are already on record. In 2009, insurgents in the Middle East intercepted Predator drone signals due to a failure to use secure protocols, according to Cabetas. This enabled the insurgents to spy on what the Predators were spying on (via airborne video). Without secure protocols, similar attacks are possible with domestic UAVs.

And, in a 2012 case, Texas A&M college students, by invitation of Homeland Security spoofed the University drone's GPS signals, insinuating the errant location data into navigation computers, resulting in the drone's untimely collision, Cabetas notes.

"But, the scariest thing we've seen so far was accomplished by the winner of the 2012 DroneGames, a Drone programming contest. The winner created a virus that took over any Drone that came close to the infected Drone," says Cabetas. Using a single vulnerability in the homogenous firmware of the drones, an attacker could fill the skies with UAVs ready to follow his every command.

And, in a couple of years, drones will be standard components of physical penetration testing, corporate espionage and hacker attacks, according to Cabetas. "Attackers could take high resolution photos and videos in windows (looking for passwords on sticky notes and other sensitive data). They'll be able to plant high fidelity microphones for eavesdropping on the outside of sensitive rooms (conference rooms, CEO offices)," Cabetas asserts.

Reference

http://www.lovemytool.com/files/vulnerabilities-threats-and-attacks-chapter-one-7.pdf

https://www.thehaguesecuritydelta.com/media/com_hsd/report/57/document/4aa6-3786enw.pdf

https://www.lifewire.com/brief-history-of-malware-153616

https://www.lifewire.com/what-is-a-boot-sector-2625815

http://whatis.techtarget.com/definition/macro

https://usa.kaspersky.com/internet-security-center/definitions/macro-virus#.WFNa2Rt95PY

http://www.explorehacking.com/2011/01/batch-files-art-of-creating-viruses.html

https://www.webroot.com/in/en/home/resources/tips/pc-security/security-what-are-bots-botnets-and-zombies

http://www.howtogeek.com/180615/keyloggers-explained-what-you-need-to-know/

https://www.cs.arizona.edu/~collberg/Teaching/466-566/2012/Resources/presentations/2012/topic5-final/report.pdf

http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/the-evolution-of-ransomware.pdf

https://admin-ahead.com/blog/ping-flood-attack/

https://usa.kaspersky.com/internet-security-center/definitions/smurf-attack#.WFT-aht95PY

https://www.symantec.com/content/en/us/enterprise/white_papers/b-advanced_persistent_threats_WP_21215957.en-us.pdf

https://sharkscale.wordpress.com/2016/02/06/defending-against-stuxnet/

http://www.howtogeek.com/141944/htg-explains-why-windows-has-the-most-viruses/

http://www.howtogeek.com/217043/xprotect-explained-how-your-macs-built-in-anti-malware-works/

https://www.fortinet.com/content/dam/fortinet/assets/solution-guides/Why-Do-You-Need-Sandboxing.pdf

http://www.zdnet.com/article/bromium-a-virtualization-technology-to-kill-all-malware-forever/

https://www.us-cert.gov/sites/default/files/publications/malware-threats-mitigation.pdf

https://zeltser.com/mastering-4-stages-of-malware-analysis/

https://www.corelan.be/index.php/2009/07/19/exploit-writing-tutorial-part-1-stack-based-overflows/

http://techtalk.gfi.com/threats-steganography/

http://www.windowsecurity.com/articles-tutorials/windows_os_security/Alternate_Data_Streams.html

https://www.recordedfuture.com/threat-actor-types/

http://online-passport.info/comsecpriv/?page_id=39

https://www.f-secure.com/en/web/labs_global/exploit-kits

https://www.webroot.com/blog/2016/03/31/malware-service-easy-gets/

https://securityintelligence.com/cybercrime-as-a-service-poses-a-growing-challenge/

http://www.csoonline.com/article/2134265/network-security/the-internet-of-things--top-five-threats-to-iot-devices.html